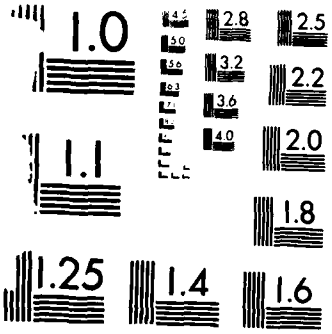
AD-A165 346 ADA (TRADEMARK) TRAINING CURRICULUM INSTRUCTOR'S COURSE 1/3
S500 ADVANCED LANGUAGE MODULES(U) SOFTECH INC WALTHAM
MA 1986 DAAB07-85-C-K506

UNCLASSIFIED F/G 5/9 NL



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

1

1986

Ada® Training Curriculum

DTIC FILE COPY

AD-A165 346

86 3 11 150

Instructor's Course
S500
Advanced Language Modules

DTIC
ELECTE

U.S. Army Communications-Electronics Command
(CECOM)

Contract DAAB07-85-C-K506

Prepared By:

SOFTECH, INC.
460 Totten Pond Road
Waltham, MA 02154

•Ada is a registered trademark of the U.S. Government, Ada Joint Program Office

*Approved For Public Release/Distribution Unlimited

SOFTECH

INSTRUCTOR'S COURSE MODULE (S500)

931/D - ADVANCED LANGUAGES MODULES

This document has been approved
for public release and sale; its
distribution is unlimited.

Accession For
NTIS GRA&I ☒
DTIC TAB □
Unannounced □
Justification
By
Distribution/
Availability Codes
Dist | Avail and/or Special
A-1

QUALITY INSPECTED 3

This INSTRUCTOR'S COURSE consists of the following volumes:

ADVANCED LANGUAGE MODULES

(1) OVERVIEW;

(2) L305 - ADVANCED TOPICS;

(3) L401 - REAL TIME SYSTEMS IN Ada;

4. L402 - USING THE Ada LANGUAGE REFERENCE MANUAL. Keywords:

ADA programming language, training, instruction manuals.

INSTRUCTOR'S NOTES

- ALLOW 10 MINUTES FOR THIS SECTION

Section 1

OVERVIEW

INSTRUCTOR NOTES

- EMPHASIZE THAT B.S. IN COMPUTER SCIENCE IS NOT ASSUMED.

OVERVIEW OF L305

- GOALS
  - THOROUGH MASTERY OF MODULARITY AND ENCAPSULATION
  - EXPOSE STUDENTS TO COMPLEX ALGORITHMS AND DATA STRUCTURES
  - TEACH Ada FEATURES RELATED TO ABOVE GOALS

- GOALS DO NOT INCLUDE
  - PROVIDING THOROUGH COVERAGE OF ANY PARTICULAR CLASS OF DATA STRUCTURES AND ALGORITHMS, E.G., SORTING AND SEARCHING

- STUDENT BACKGROUND
  - BASIC Ada PROGRAMMING (L202)
  - PROGRAMMING METHODOLOGY (M203)
  - STUDENTS ARE NOT ASSUMED TO HAVE B.S. IN COMPUTER SCIENCE OR EQUIVALENT

- MODULE OVERVIEW (5 DAYS/10 DAYS)
  - THIS MODULE TEACHES MODERN ABSTRACTION CONCEPTS AND RELATED FACILITIES OF Ada. IT STRESSES KEY CONCEPTS OF ABSTRACTION AND INFORMATION HIDING IN THE CONTEXT OF ADVANCED PROGRAMMING TECHNIQUES

INSTRUCTOR NOTES

- EMPHASIZE MIXTURE OF CLASS POSSIBLE
  - EXPERIENCED REAL TIME PROGRAMMERS
  - PROGRAMMER WITH LITTLE, IF ANY, EXPOSURE TO CONCURRENT PROGRAMMING

OVERVIEW OF L401

- GOALS
  - TEACH TASKING FEATURES OF Ada
  - INTRODUCE CONCEPTS OF CONCURRENT PROGRAMMING/REAL TIME PROGRAMMING
  - TEACH EXPERIENCED REAL TIME PROGRAMMERS HOW TO USE Ada TASKING FEATURES TO SOLVE PROBLEMS WITH WHICH THEY ARE FAMILIAR
  - TEACH WHEN AND HOW TO IMPROVE PROGRAM PERFORMANCE

- GOALS DO NOT INCLUDE
  - TEACHING ABOUT SPECIFIC IMPLEMENTATIONS OF Ada
  - TEACHING ABOUT SPECIFIC TARGET COMPUTERS

- STUDENT BACKGROUND
  - ADVANCED Ada TOPICS (L305)
  - STUDENTS ARE NOT ASSUMED TO HAVE CONCURRENT PROGRAMMING BACKGROUND

- MODULE OVERVIEW (5 DAYS)
  - THIS MODULE TEACHES CONCURRENT PROGRAMMING IN Ada WITH EMPHASIS ON REAL TIME PROGRAMMING. IT ALSO TEACHES WHEN AND HOW PROGRAMS (SEQUENTIAL OR CONCURRENT) CAN BE TUNED TO IMPROVE PERFORMANCE.

INSTRUCTOR NOTES

OVERVIEW OF L402

- GOALS
  - DEFINE LANGUAGE TERMS IN LRM AND WHERE TERMS ARE DISCUSSED
  - FAMILIARIZE STUDENTS WITH SUBTLE SEMANTIC ISSUES AND HOW TO RESOLVE LANGUAGE ISSUES IN GENERAL

- GOALS DO NOT INCLUDE
  - TEACH PROGRAMMING
  - TEACH Ada
  - TEACH EVERY DETAIL IN LRM

- STUDENT BACKGROUND
  - ADVANCED Ada TOPICS (L305)

- MODULE OVERVIEW (2 DAYS)
  - THIS MODULE TEACHES HOW TO USE THE LRM. STUDENTS WILL UNDERSTAND HOW TO FIND SECTIONS OF THE LRM PERTAINING TO A PROBLEM OR QUESTION AND HOW TO INTERPRET THESE SECTIONS.

INSTRUCTOR NOTES

Section 2

L305

ADVANCED TOPICS

INSTRUCTOR NOTES

- THE FOUR REFERENCES ON THIS PAGE AND THE NEXT SHOULD BE CONSIDERED MUST READING FOR AN L305 INSTRUCTOR

    - POSSIBLE EXCEPTION IS Data Structures and Algorithms IF INSTRUCTOR HAS STRONG BACKGROUND IN DATA STRUCTURES AND ALGORITHMS

    - THESE REFERENCES HAVE BEEN CHOSEN TO GIVE AN L305 INSTRUCTOR A STRONG BACKGROUND IN (NOT NECESSARILY Ada) TOPICS COVERED IN THIS MODULE

L305 - INSTRUCTOR'S BIBLIOGRAPHY

- Ada Case Studies II CONTRACT NO. DAAB07-83-C-K514, CECOM, JANUARY 1984, ALSO TO APPEAR AS Ada In Practice, Springer-Verlag Publishing Co.
  - 15 CASE STUDIES COVERING
    - NAMING CONVENTIONS
    - TYPES
    - CODING PARADIGMS
    - EXCEPTIONS
    - PROGRAM STRUCTURE
  - ANALYZES CONTRACTOR'S FIRST EXPERIENCE WRITING AN Ada COMMUNICATIONS PROGRAM CALLED THE MESSAGE SWITCH
    - PORTION OF EXISTING CODE SELECTED
    - PURPOSE OF CODE EXPLAINED
    - POSSIBLE SHORTCOMINGS IN CODE POINTED OUT
    - SUGGESTIONS FOR MODIFICATIONS ARE MADE THAT COULD IMPROVE EFFICIENCY, READABILITY, OR CONSISTENCY
    - REVISED CODE GENERALLY INCLUDED
- Data Structures and Algorithms, A.V. Aho, J.E. Hopcroft, J.D. Ullman, Addison-Wesley Publishing Co., 1983
  - MOST OF THE DATA STRUCTURES AND ALGORITHMS DISCUSSED IN THIS MODULE ARE DISCUSSED IN DETAIL IN THIS TEXT
  - EXAMPLES ARE WRITTEN IN PASCAL
  - IN-DEPTH COVERAGE SHOULD MAKE INSTRUCTORS MORE COMFORTABLE IN TEACHING THIS MODULE

- REFERENCE 2

  - THIS IS NOT A SUGGESTION THAT AN L305 INSTRUCTOR SHOULD DISCUSS IMPLEMENTATION OF GENERICS OR EFFICIENCY OF THEIR USE. IN FACT, THE STUDENTS SHOULD BE URGED TO STAY AWAY FROM ANY SUCH DISCUSSIONS. A "GOOD" IMPLEMENTATION WILL IMPLEMENT GENERICS IN A WAY THAT WILL NOT CAUSE STUDENTS TO QUESTION THEIR EFFICIENCY.

  - THIS REFERENCE IS INCLUDED FOR THE L305 INSTRUCTOR'S BENEFIT IN CASE HE/SHE IS FORCED TO DISCUSS EFFICIENCY.

L305 - INSTRUCTOR'S BIBLIOGRAPHY - continued

- Four Uses For Derived Types and a Complication, N.H. Cohen, IEEE Computer Society 1984 Conference on Ada Applications and Environments, OCTOBER 1984, 106-115
  - IN-DEPTH COVERAGE OF DERIVED TYPES
    - USES
    - PROBLEMS
  - L305 INSTRUCTORS SHOULD READ THIS PAPER BEFORE TEACHING L305 DERIVED TYPES

- Implementation Implications of Ada Generics, G. Broy Ada Letters, 3(2), SEPTEMBER 1983, 62-71
  - DISCUSSES POSSIBLE WAYS TO IMPLEMENT GENERIC INSTANTIATIONS
  - STRENGTHS AND WEAKNESSES OF VARIOUS APPROACHES
  - PROVIDES L305 INSTRUCTOR WITH BACKGROUND TO EXPLAIN, IF NECESSARY, HOW INSTANTIATIONS MIGHT BE IMPLEMENTED EFFECTIVELY

INSTRUCTOR NOTES

L305 - GENERAL COMMENTS

- IN-CLASS EXERCISES ARE SCATTERED THROUGHOUT THE COURSE
- LAB EXERCISES ARE DISCUSSED AT THE POINTS WHERE THEY COULD BE ASSIGNED IN TEACHING THE COURSE. IT IS ASSUMED THAT THEY WILL BE OMITTED FROM A ONE-WEEK VERSION.
- SPECIAL CONSIDERATIONS ARE NOTED WHERE APPLICABLE, BASED ON PRIOR EXPERIENCE TEACHING THE MATERIAL
- EACH MAJOR SUBSECTION IS DISCUSSED, ITS MAIN POINTS AND ITS RELATION TO THE COURSE OVERALL

INSTRUCTOR NOTES

- GIVE INSTRUCTORS IN TRAINING AN OVERVIEW OF WHAT IS COVERED IN L305 PART I AND WHAT ITS OBJECTIVES ARE

- EMPHASIZE THAT SECTION 1 IS REVIEW. STRESS THIS NOW AND IN COVERAGE OF SECTION 1.

- TARGET TEACHING TIME IS IN PARENTHESES

  - INCLUDED TO GIVE INSTRUCTORS IN TRAINING AN IDEA OF HOW MUCH TIME IS DEVOTED TO VARIOUS TOPICS

  - EMPHASIZE THAT TIME IS TARGET. MAY VARY DEPENDING ON CLASS NEEDS/BACKGROUND

- L305 INSTRUCTORS SHOULD EMPHASIZE THAT SECTION 1 OF L305 IS JUST A REVIEW

PART I - BASIC STRUCTURING FEATURES

FOCUS OF UNIT:

- REVIEW FUNDAMENTAL PROGRAM AND DATA STRUCTURING FEATURES OF Ada
- LAY GROUNDWORK FOR REST OF COURSE, WHICH USES THESE FEATURES HEAVILY
- INTRODUCE RECURSION, PRESENTING REASONS FOR ITS USE
- DISCUSS MUTUAL RECURSION

SECTIONS:

1. REVIEW OF PACKAGES AND NONSCALAR TYPES (1:50)

   A. PACKAGES

   B. ARRAYS

   C. RECORDS

   D. ACCESS TYPES

2. RECURSIVE PROGRAMMING (:45)

INSTRUCTOR NOTES

THIS SECTION IS PRIMARILY INTENDED AS A REVIEW OF PACKAGE BASICS. THE MATERIAL SHOULD BE FAMILIAR TO THE STUDENTS THE INSTRUCTORS IN TRAINING WILL BE TEACHING. THE GENERAL IDEA IS TO "GET EVERYBODY TO THE SAME LEVEL."

THE 3RD BULLET OF THE MAIN MESSAGES IS THE KEY. POINT OUT TO THE INSTRUCTORS IN TRAINING THAT THIS IS A RECURRING AND HEAVILY STRESSED THEME.

TELL THE INSTRUCTORS IN TRAINING THAT THE SUBTOPICS ARE ILLUSTRATED WITH SMALL BUT COMPLETE EXAMPLES.

TELL THE INSTRUCTORS IN TRAINING THAT THE SUBSECTION ON use CLAUSES ALSO DISCUSSES STYLE ISSUES, SPECIFICALLY WHEN IT IS APPROPRIATE TO USE IT.

THE WORD REVIEW IS UNDERLINED TO REMIND THE INSTRUCTORS IN TRAINING THAT THEY ARE TO GIVE THEIR L305 STUDENTS A REVIEW, NOT TEACH THEM NEW MATERIAL.

SECTION 1 - REVIEW

PACKAGES

SUMMARY OF MAIN POINTS COVERED:

- PACKAGE SPECIFICATION PROVIDES ALL INTERFACE INFORMATION
- NAMING OF ENTITIES
- WITH CLAUSE
- USE CLAUSE
- RENAMING DECLARATIONS

MAIN MESSAGES:

- PACKAGES ARE AN IMPORTANT DESIGN TOOL
- PACKAGES PROVIDE A WAY OF GROUPING RELATED ENTITIES
- PACKAGES DISTINGUISH INTERFACE FROM IMPLEMENTATION

INSTRUCTOR NOTES

SECTION 1 - REVIEW

PACKAGES - Continued

SUBTOPICS:

- REVIEW SYNTAX OF PACKAGES

  - SPECIFICATION
    - VISIBLE PART ONLY
    - PRIVATE PART FIRST DISCUSSED IN L305 PART III
    - SUBPROGRAM SPECIFICATION
  - BODY
    - DECLARATIVE PART
    - INITIALIZATION

- REVIEW NAMING OF ENTITIES DECLARED IN PACKAGE SPECIFICATION

  - INSIDE PACKAGE: BY ITS IDENTIFIER
  - OUTSIDE PACKAGE: BY EXPANDED NAME

- REVIEW AND EMPHASIZE

  - PACKAGE SPECIFICATION AS INTERFACE
  - PACKAGE BODY AS IMPLEMENTATION

- REVIEW USING PACKAGES

  - WITH CLAUSE
  - USE CLAUSE
  - RENAMING DECLARATIONS

INSTRUCTOR NOTES

SECTION 1 - REVIEW

PACKAGES - Continued

SPECIAL CONSIDERATIONS:

- EVEN THOUGH THIS IS AN OVERVIEW, L305 INSTRUCTORS SHOULD FEEL FREE TO EMPHASIZE AGAIN AND AGAIN THAT THE PACKAGE SPECIFICATION PROVIDES THE INTERFACE:

    - L305 INSTRUCTORS SHOULD EMPHASIZE THAT ONLY INTERFACE INFORMATION SHOULD APPEAR IN THE PACKAGE SPECIFICATION

INSTRUCTOR NOTES

- AN L305 INSTRUCTOR SHOULD EMPHASIZE
  - UNCONSTRAINED ARRAY TYPES
  - AGGREGATES
  - SLICES AND CATENATION
  - ATTRIBUTES

  THESE ARE FEATURES STUDENTS WILL HAVE FORGOTTEN IF THEY HAVEN'T USED THEM MUCH

- THE WORD REVIEW IS UNDERLINED SO THAT THE INSTRUCTORS IN TRAINING REALIZE THEY ARE TO GIVE L305 STUDENTS A REVIEW, NOT AN INTRODUCTION

SECTION 1 - REVIEW

ARRAYS

SUMMARY OF MAIN POINTS COVERED:

- REVIEW THE DECLARATION AND USE OF ARRAY TYPES

MAIN MESSAGES:

- ARRAY TYPES FORM THE BASIS FOR MANY ABSTRACTIONS, I.E., THEY PROVIDE THE UNDERLYING IMPLEMENTATION

SUBTOPICS:

- <u>REVIEW</u> SYNTAX OF ARRAY TYPES
    - UNCONSTRAINED TYPES
    - CONSTRAINED TYPES
- <u>REVIEW</u> ARRAY AGGREGATES
- <u>REVIEW</u> ARRAY OBJECT DECLARATIONS
    - WITH INITIAL VALUES
    - ONE-OF-A-KIND ARRAYS (ANONYMOUS ARRAY TYPE)
- <u>REVIEW</u> OPERATIONS
    - COMPONENT SELECTION
    - ASSIGNMENT
    - COMPARISON
    - SLICES
    - CATENATION
    - ATTRIBUTES

INSTRUCTOR NOTES

SECTION 1 - REVIEW

ARRAYS - Continued

SPECIAL CONSIDERATIONS:

- AN EASY TRAP TO FALL INTO IS TO GIVE AN INTRODUCTION RATHER THAN A REVIEW

INSTRUCTOR NOTES

- EVEN THOUGH THIS IS A REVIEW, L305 INSTRUCTORS SHOULD BE PREPARED TO SPEND ADDITIONAL TIME ON RECORDS WITH DISCRIMINANTS. HOWEVER, IF TOO MUCH TIME IS REQUIRED, THE L305 INSTRUCTOR SHOULD BE READY TO DO SO OFF-LINE.

- AGAIN, WARN THE INSTRUCTORS IN TRAINING ABOUT THE TRAP FROM 2-7

SECTION 1 - REVIEW

RECORDS

SUMMARY OF MAIN POINTS COVERED:

- DECLARATION AND USE OF RECORD TYPES
- DISCRIMINANTS

MAIN MESSAGES:

- RECORDS ARE A FUNDAMENTAL DATA STRUCTURING FEATURE
- THIS SECTION IS A REVIEW OF THE BASIC FEATURES

SUBTOPICS:

- REVIEW SYNTAX OF RECORD TYPES
  - COMPONENT TYPES
  - DISCRIMINANTS AND VARIANTS
  - VARIANTS (INCLUDING NESTED VARIANTS)
  - INITIAL VALUES FOR COMPONENTS
- REVIEW RECORD AGGREGATES
  - POSITIONAL
  - NAMED
  - MIXED
  - OTHERS
- REVIEW RECORD OBJECT DECLARATIONS
  - WITH INITIAL VALUES
  - CONSTRAINED/UNCONSTRAINED
- REVIEW OPERATIONS
  - COMPONENT SELECTION
  - ASSIGNMENT
  - COMPARISON
  - ATTRIBUTES: 'CONSTRAINED

INSTRUCTOR NOTES

REMIND THE INSTRUCTORS IN TRAINING THAT THEY ARE STILL PRESENTING REVIEW MATERIAL.

SECTION 1 - REVIEW

ACCESS TYPES

SUMMARY OF MAIN POINTS COVERED:

- REVIEW OF THE DECLARATION AND USE OF NONRECURSIVE ACCESS TYPES

MAIN MESSAGES:

- ACCESS TYPES ARE USED IN IMPLEMENTING MANY ABSTRACTIONS

SUBTOPICS:

- REVIEW SYNTAX OF ACCESS TYPES
  - NULL VALUE
- REVIEW ALLOCATORS
  - CONSTRAINTS
  - INITIAL VALUE
- REVIEW OBJECT DECLARATIONS
- REVIEW OPERATIONS
  - ACCESSING THE DESIGNATED TYPE
  - ASSIGNMENT
  - EQUALITY
  - ATTRIBUTES

SPECIAL CONSIDERATIONS:

- AVOID DISCUSSING RECURSIVE TYPES - THEY ARE TREATED IN DEPTH IN PART II
- EMPHASIZE THE DISTINCTION BETWEEN AN ACCESS VALUE AND WHAT IT POINTS TO

INSTRUCTOR NOTES

SECTION 2 - RECURSIVE SUBPROGRAMS

SUMMARY OF MAIN POINTS COVERED:

- DEFINE AND ILLUSTRATE RECURSION

MAIN MESSAGES:

- Ada ALLOWS BOTH SIMPLE AND MUTUAL RECURSION

SUBTOPICS:

- SIMPLE RECURSION
- MUTUAL RECURSION
- EXAMPLES

INSTRUCTOR NOTES

- MAKE SURE THE INSTRUCTORS IN TRAINING REALIZE THAT THE REMAINING MATERIAL IS NEW COURSE MATERIAL

- BULLET #3
  - L305 INSTRUCTORS SHOULD RAISE THESE ISSUES IN CASE THEY HAVEN'T OCCURRED TO SOME L305 STUDENTS

- DEPENDING ON THE BACKGROUND OF AN L305 CLASS, AN L305 INSTRUCTOR SHOULD
  - GO INTO DETAIL ABOUT RECURSION IF MOST OF THE CLASS HAS HAD LITTLE EXPOSURE TO RECURSION
  - JUST BRIEFLY DISCUSS RECURSION IN Ada, INCLUDING MUTUAL RECURSION, IF THE CLASS UNDERSTANDS RECURSION

SPECIAL CONSIDERATIONS:

- THIS SECTION BEGINS THE NEW L305 MATERIAL

- RECURSION MAY BE NEW TO MANY L305 STUDENTS
  - EARLY LANGUAGES DO NOT SUPPORT RECURSION
  - FORTRAN, COBOL (ALSO ASSEMBLER)

- L305 STUDENTS MAY WONDER ABOUT
  - LOCAL VARIABLES
    - ARE OLD VALUES DESTROYED?
    - HOW ARE NEW VARIABLES CREATED?
  - SUBPROGRAM CODE
    - IS A NEW COPY OF THE CODE CREATED FOR EACH ALL?
    - HOW CAN ONE COPY OF THE CODE BE USED?
    - HOW CAN RECURSION TERMINATE?

- EMPHASIZE RECURSION USED TO SOLVE EASIER VERSIONS OF SAME PROBLEM
  - EXAMPLE:
    - TRAVERSE A TREE BY TRAVERSING EACH CHILD'S SUBTREE RECURSIVELY

- L305 STUDENTS NEED TO REALIZE THAT TERMINATION CONDITIONS FOR RECURSIVE SUBPROGRAMS MUST BE CAREFULLY THOUGHT OUT
  - SPEND EXTRA TIME, IF NEEDED, TO GET STUDENTS TO UNDERSTAND IMPORTANCE OF TERMINATION CONDITIONS

- FOR MUTUAL RECURSION
  - MAKE SURE CLASS UNDERSTANDS THE PROBLEM WITH CALLING A SUBPROGRAM BEFORE ITS DECLARATION
  - MAKE SURE CLASS UNDERSTANDS HOW SUBPROGRAM SPECIFICATION IS USED AND HOW IT SOLVES THE PROBLEM

INSTRUCTOR NOTES

- ASSIGNING THIS EXERCISE WILL HELP AN L305 INSTRUCTOR RECOGNIZE A STUDENT WHOSE L202 BACKGROUND IS WEAK

PART I : EXERCISE

- EXERCISE 1 - PACKAGE DESIGN
  - REVIEWS CONCEPTS FROM L202
  - GOOD WARM-UP SO SHOULD BE ASSIGNED
  - TESTS UNDERSTANDING OF FUNDAMENTAL PRINCIPLES UNDERLYING PACKAGES
    - DISTINCTION BETWEEN INTERFACE AND IMPLEMENTATION
  - FAIRLY EASY
    - ONLY CATCH IS NEED FOR FIXED POINT TYPE REQUIRED
    - REMINDS STUDENTS OF TYPE CONVERSIONS
  - MAY BE ASSIGNED AFTER SECTION 1

INSTRUCTOR NOTES

- GIVE INSTRUCTORS IN TRAINING AN OVERVIEW OF WHAT IS COVERED IN PART II OF L305

PART II - FUNDAMENTAL DATA STRUCTURES

FOCUS OF UNIT:

- DISCUSS FUNDAMENTAL DATA STRUCTURES AND DEVELOP BASIC OPERATIONS
- MAKE STUDENTS AWARE OF ABSTRACTION BEHIND EXAMPLES
- EMPHASIZE THAT AN ABSTRACTION CAN BE IMPLEMENTED IN MANY DIFFERENT WAYS

SECTIONS:

3. SETS USING BOOLEAN ARRAYS (:25)
4. LINEAR LISTS (1:10)
5. LINKED LISTS AND RECURSIVE TYPES (1:15)

SPECIAL CONSIDERATIONS:

- AN L305 INSTRUCTOR MAY NEED TO TUNE THIS PART TO L305 CLASS BACKGROUND
    - IF L305 CLASS HAS STRONG BACKGROUND IN DATA STRUCTURES, THEN EMPHASIZE HOW TO DO IT IN Ada
    - IF L305 CLASS DOES <u>NOT</u> HAVE GOOD BACKGROUND IN DATA STRUCTURES, THEN FAIR AMOUNT OF TIME MUST BE SPENT ON DATA STRUCTURES THEMSELVES
        - POSSIBLE TO BYPASS MULTILISTS IF ADDITIONAL TIME IS NEEDED

INSTRUCTOR NOTES

SECTION 3 - SETS USING BOOLEAN ARRAYS

SUMMARY OF MAIN POINTS COVERED:

- INTRODUCE SETS WITH ELEMENTS BELONGING TO A FINITE UNIVERSE

MAIN MESSAGES:

- SETS CAN BE USED TO CHARACTERIZE MANY PROBLEMS

SUBTOPICS:

- OPERATIONS ON SETS
    - MEMBERSHIP
    - UNION
    - INTERSECTION
    - DIFFERENCE
    - EXTRACTION
    - INSERTION
    - ITERATORS : FOR EACH ELEMENT

INSTRUCTOR NOTES

- Ada CASE STUDIES REFERS TO

  <u>Ada DESIGN METHODS AND TRAINING SUPPORT CASE STUDIES II</u>

  THIS DOCUMENT IS DESCRIBED IN THE BIBLIOGRAPHY AT THE BEGINNING OF THIS SECTION.

- URGE INSTRUCTORS IN TRAINING TO READ THE CASE STUDIES

SECTION 3 - SETS USING BOOLEAN ARRAYS - Continued

SPECIAL CONSIDERATIONS:

- THE MESSAGE RELAY PROBLEM
  - USE OF SETS FIRST ILLUSTRATED IN MATHEMATICAL/Ada NOTATION
    - ALLOWS STUDENTS TO SEE SET ABSTRACTION WITHOUT Ada CODING DETAILS
  - IMPLEMENTATION IN Ada IS GIVEN NEXT
  - DO <u>NOT</u> GO INTO DETAIL ABOUT THE PROBLEM ITSELF
    - ENOUGH DETAILS TO UNDERSTAND WHY SETS ARE USED
    - DETAILS IN INSTRUCTOR'S NOTES FOR L305 INSTRUCTOR'S BENEFIT ONLY

- SOME LANGUAGES PROVIDE SETS
  - EXAMPLES: Pascal, MODULA-2, Euclid, SETL
  - SOME CRITICAL OF Ada DUE TO LACK OF SETS
    - Pascal-LIKE SETS CAN BE IMPLEMENTED USING
      - PACKAGES
      - OVERLOADING
    - <u>Ada CASE STUDIES II</u> PROVIDE Pascal-LIKE IMPLEMENTATION OF SETS
    - L305 EXERCISE ASKS STUDENTS TO IMPLEMENT Pascal-LIKE SOLUTION
  - BE PREPARED TO RESPOND TO QUESTIONS ABOUT SETS ASKED BY PASCAL PROGRAMMERS
    - READ THE Ada CASE STUDIES II SECTION (2.2)

INSTRUCTOR NOTES

SECTION 4 - LINEAR LISTS

SUMMARY OF MAIN POINTS COVERED:

- INTRODUCE LISTS - FIXED AND VARIABLE LENGTH
- DEFINE BASIC LIST OPERATIONS - ABSTRACTLY, THEN Ada CODE
- STACKS AND QUEUES

MAIN MESSAGES:

- LINEAR LISTS CAN BE USED TO BUILD MORE SPECIALIZED DATA STRUCTURES
- STACKS AND QUEUES ARE SPECIAL CASES OF LISTS

SUBTOPICS:

- LISTS IMPLEMENTED USING ONE-DIMENSIONAL ARRAYS
  - FIXED-LENGTH
  - VARIABLE LENGTH
- BASIC LIST OPERATIONS
  - EXAMINE ITEM AT POSITION P
  - MODIFY ITEM AT POSITION P
  - PERFORM OPERATION FOR EACH ITEM IN LIST
- ADDITIONAL OPERATIONS FOR VARIABLE-LENGTH LIST
  - RETURN CURRENT LENGTH
  - INSERT ITEM AT POSITION P
  - DELETE ITEM AT POSITION P

INSTRUCTOR NOTES

- BULLET #2, SPECIAL CONSIDERATION
  - THE STRAIGHTFORWARD QUEUE IMPLEMENTATION IS TO ADD ITEMS AS

    ```
    QUEUE (LENGTH) : = ITEM;
    LENGTH : = LENGTH + 1;
    ```

    AND DELETE ITEMS AS

    ```
    LENGTH : = LENGTH - 1;
    QUEUE (1 .. LENGTH) : = QUEUE (2 .. LENGTH + 1);
    ```

    HOWEVER DELETING ELEMENTS THIS WAY IS INEFFICIENT AND A MORE REASONABLE IMPLEMENTATION MUST BE GIVEN. OTHERWISE

    - L305 STUDENTS LEARNING ABOUT QUEUES FOR THE FIRST TIME MIGHT MISTAKENLY THINK THAT LINEAR IMPLEMENTATION OF QUEUES ARE INEFFICIENT
    - L305 STUDENTS FAMILIAR WITH QUEUES MIGHT BECOME SUSPECT OF THE TRAINING MATERIAL

- LAST BULLET
  - THERE ARE SEVERAL POSSIBLE VARIATIONS OF CIRCULAR LIST IMPLEMENTATIONS OF QUEUES. EACH HAS ITS OWN "TRICKY" WAY OF INDICATING WHEN A QUEUE IS FULL AND WHEN IT IS EMPTY

SUBTOPICS: (Continued)

- STACKS
  - TOP OF STACK
  - OPERATIONS
    - PUSH
    - POP
    - TEST FOR EMPTY STACK
- QUEUES
  - OPERATIONS
    - INSERT AT BACK
    - DELETE FRONT ITEM
    - EXAMINE FRONT ITEM
    - TEST FOR EMPTY/FULL QUEUE
  - CIRCULAR LIST IMPLEMENTATION

SPECIAL CONSIDERATIONS:

- EMPHASIZE THAT INFINITELY MANY OPERATIONS ARE POSSIBLE ON THE PHYSICAL DATA BUT ONLY A HANDFUL ARE MEANINGFUL ON THE CONCEPTUAL DATA
- STACK IMPLEMENTATION IS LESS COMPLEX THAN QUEUE IMPLEMENTATIONS FOR EFFICIENCY
  - NORMALLY DO NOT TALK ABOUT EFFICIENCY
  - STRAIGHTFORWARD QUEUE IMPLEMENTATION IS TOO INEFFICIENT
- STACK PROBLEM AND QUEUE PROBLEM EMPHASIS SHOULD BE ON
  - ABSTRACTLY DEFINED OPERATIONS
  - STUDENTS NEED TO SEE ABSTRACTION TO START THINKING IN TERMS OF ABSTRACTIONS RATHER THAN CODE

  Ada CODE IS GIVEN LATER TO SHOW HOW ABSTRACTION CAN BE IMPLEMENTED
- SEVERAL QUEUE IMPLEMENTATIONS GIVEN
  - STRAIGHTFORWARD BUT EXTREMELY INEFFICIENT IMPLEMENTATION
  - CIRCULAR LIST IMPLEMENTATION
    - SEVERAL VARIATIONS
    - BE PREPARED TO DISCUSS ALL

INSTRUCTOR NOTES

SECTION 5 - LINKED LISTS AND RECURSIVE TYPES

SUMMARY OF MAIN POINTS COVERED:

- INTRODUCE RECURSIVE TYPES AND THEIR USE
- DISCUSS ADVANTAGES AND DISADVANTAGES OF VARIATIONS IN LINKED LIST IMPLEMENTATIONS
- COMPARE LINKED LISTS WITH LINEAR LISTS
- STACKS AND QUEUES RE-IMPLEMENTED AS LINKED LISTS
- OVERVIEW OF MULTILISTS

MAIN MESSAGES:

- MANY COMMON DATA TYPES ARE DEFINED RECURSIVELY
    - LINKED LISTS ARE THE PRIMARY EXAMPLE
- THERE ARE MANY WAYS TO IMPLEMENT A SINGLE DATA ABSTRACTION

SUBTOPICS:

- INCOMPLETE TYPE DECLARATIONS
    - SUMMARY : DECLARING RECURSIVE TYPES
- IMPLEMENTATION OF LIST OPERATIONS USING LINKED LISTS
    - USE/REPLACE VALUE AT POSITION P
    - PERFORM OPERATION FOR EACH ITEM IN THE LIST
    - INSERT VALUE BEFORE/AFTER POSITION N
    - DELETE VALUE FROM NON-EMPTY LIST
- VARIATIONS
    - DUMMY LIST CELL
    - DOUBLY LINKED LIST
- STACKS AND QUEUES USING LINKED LISTS
- OVERVIEW OF MULTILISTS
    - APPLICATIONS

INSTRUCTOR NOTES

- AN L305 INSTRUCTOR SHOULD EMPHASIZE THAT THERE ARE MANY WAYS TO IMPLEMENT LISTS. WHICH WAY IS BEST IS A FUNCTION OF THE INTENDED USE. THE CITED TEXT DISCUSSES THIS.

- THE CITED TEXT (AUTHORS: AHO, HOPCROFT AND ULLMAN) COVERS MOST OF THE MATERIAL COVERED IN PART II. EXAMPLES ARE WRITTEN IN Pascal.

SECTION 5 - LINKED LISTS AND RECURSIVE TYPES - Continued

SPECIAL CONSIDERATIONS:

- IF L305 STUDENTS ARE ALREADY FAMILIAR WITH RECURSIVE TYPES THEN DISCUSS INCOMPLETE TYPE DEFINITIONS BUT DON'T SPEND TOO MUCH TIME ON INTRODUCING RECURSIVE TYPES

- MAKE SURE CLASS UNDERSTANDS INCOMPLETE TYPE DEFINITIONS

- EMPHASIZE MANY WAYS TO IMPLEMENT A SINGLE DATA ABSTRACTION

- MULTILIST IMPLEMENTATION IS AN OVERVIEW ONLY
  - EXAMPLE: DATABASE, SPARSE MATRIX
  - OVERVIEW ONLY: DO NOT GET INVOLVED IN DETAILS

- GOOD REFERENCE FOR INSTRUCTORS
  - DATA STRUCTURES AND ALGORITHMS

INSTRUCTOR NOTES

PART II - EXERCISES

- EXERCISE 2 - LIST MANIPULATION
  - STUDENTS REQUIRED TO WRITE A PACKAGE PROVIDING LIST OF INTEGER VALUES
    - MUST USE DOUBLY LINKED-LIST
    - MUST USE DUMMY LIST CELL
  - TIME CONSUMING PROBLEM
  - SUPPLEMENTS COURSE COVERAGE OF DOUBLY LINKED LISTS
  - REQUIRES ORIGINAL ANALYTIC THOUGHT ABOUT POINTER MANIPULATION
  - MAY BE ASSIGNED AFTER SECTION 5

WARNING: EITHER THIS EXERCISE OR EXERCISE 3 SHOULD BE ASSIGNED BUT NOT BOTH

EXERCISE 3 ALSO COVERS LIST MANIPULATION

ASSIGNING BOTH WILL QUICKLY CAUSE STUDENTS TO GROW TIRED OF LIST MANIPULATION

INSTRUCTOR NOTES

- GIVE INSTRUCTORS IN TRAINING AN OVERVIEW OF WHAT IS COVERED IN PART III OF L305
- THESE CONCEPTS ARE USED IMPLICITLY IN THE REST OF THE MODULE
  - FORMALIZES WHAT WAS DISCUSSED IN PART II ABOUT ABSTRACTION

PART III - DATA ABSTRACTION

FOCUS OF UNIT:

- INTRODUCE CONCEPTS OF
  - ABSTRACTION
  - INFORMATION HIDING
  - ENCAPSULATION

  AND HOW THEY ARE REALIZED IN Ada
- DEFINITION OF DATA TYPE
  - PACKAGING
  - OPERATIONS
- HOW AND WHEN TO USE PRIVATE AND LIMITED PRIVATE TYPES
- USING EXCEPTIONS - DESIGNER'S ROLE VS. PROGRAMMER'S

SECTIONS:

6. DATA TYPE ENCAPSULATION (1:20)
7. PRIVATE TYPES (1:00)
8. LIMITED PRIVATE TYPES (1:15)
9. DESIGNING WITH EXCEPTIONS (:30)

INSTRUCTOR NOTES

- MAIN MESSAGES, BULLET 1 - L305 STUDENTS MUST UNDERSTAND THIS VIEW OF A DATA TYPE AND HOW Ada PACKAGES ALLOW THIS VIEW TO BE REALIZED
  - THIS SECTION DOES NOT COVER PRIVATE/LIMITED PRIVATE TYPES

- LIST EXAMPLE IS USED TO ILLUSTRATE IDEAS
  - PACKAGE DECLARATION SPECIFIES ABSTRACT TYPE List_Type
  - IMPLEMENTATION MIGHT BE
    - LINEAR LIST
    - LINKED LIST

- USER OF PACKAGE SEES ABSTRACT LIST VALUES
  - NOT ARRAY OBJECTS
  - NOT ACCESS VALUES

SECTION 6 - DATA TYPE ENCAPSULATION

SUMMARY OF MAIN POINTS COVERED:

- DISCUSS ABSTRACT DATA TYPES AND HOW THEY ARE REALIZED IN Ada
- USE OF PACKAGES
- NEED FOR PRIVACY
- ILLUSTRATE IDEAS USING LISTS

MAIN MESSAGES:

- AN ABSTRACTION FOR A DATA TYPE CONSISTS OF
  - SET OF VALUES
  - SET OF OPERATIONS (SUBPROGRAMS OF THE TYPE)
  - SET OF RELATIONSHIPS BETWEEN OPERATIONS
- Ada PROVIDES PACKAGES FOR REALIZING THIS ABSTRACTION
  - DECLARATION SPECIFIES INTERFACE
  - BODY PROVIDES IMPLEMENTATION OF ABSTRACTION
- PRIVACY IS NEEDED
  - TO ENFORCE SEPARATION OF CONCERNS
  - FOR FLEXIBILITY IN CHOICE OF IMPLEMENTATION
  - TO MAINTAIN IMPLEMENTATION INTEGRITY

INSTRUCTOR NOTES

SECTION 6 - DATA TYPE ENCAPSULATION - Continued

SUBTOPICS:

- CONCEPT OF ABSTRACT DATA TYPE
  - VALUES
  - OPERATIONS
- SUBPROGRAMS OF A TYPE
  - RELATIONSHIPS BETWEEN OPERATIONS
  - EXAMPLES
- SPECIFICATION OF AN ABSTRACTION
- IMPLEMENTATION OF AN ABSTRACTION
- NEED FOR PRIVACY
  - SEPARATION OF CONCERNS
  - IMPLEMENTATION FLEXIBILITY
  - MAINTAINING IMPLEMENTATION INTEGRITY
  - EXAMPLE USING LISTS

SPECIAL CONSIDERATIONS:

- EMPHASIZE THE INTERFACE VS. IMPLEMENTATION DISTINCTION
- WARNING: MAKE SURE STUDENTS DO NOT CONFUSE INFORMATION HIDING OR PRIVACY WITH PHYSICAL SECRECY
  - FEW STUDENTS WILL ACTUALLY BE CONFUSED ABOUT THIS
  - REACHING THOSE STUDENTS WHO MIGHT BE, WILL SAVE YOU FROM ANSWERING SEEMINGLY STRANGE QUESTIONS COMING OUT OF NOWHERE
- THE NEED FOR PRIVACY PREPARES THE L305 CLASS FOR DISCUSSION OF PRIVATE TYPES IN THE NEXT SECTION

INSTRUCTOR NOTES

- MOST OF THE EXAMPLES IN THE REST OF THE COURSE USE PRIVATE TYPES

SECTION 7 - PRIVATE TYPES

SUMMARY OF MAIN POINTS COVERED:

- INTRODUCE PRIVATE TYPES USING THE List_Type
- OPERATIONS AVAILABLE
- DISCRIMINANT PARTS

MAIN MESSAGES:

- PRIVATE TYPES ARE USED TO PREVENT DIRECT MANIPULATION OF REPRESENTATION
- IF DISCRIMINANTS ARE USED, THEY MUST BE VISIBLE TO THE USER OF THE ABSTRACTION BUT ACCESS TO DISCRIMINANTS IS READ-ONLY
- THE OPERATIONS AVAILABLE FOR PRIVATE TYPES ARE
    - PREDEFINED OPERATIONS THAT DO NOT DEPEND ON REPRESENTATION (ASSIGNMENT, EQUALITY)
    - THOSE PROVIDED BY PACKAGE DESIGNER, AS DEFINED IN PACKAGE SPECIFICATION

SUBTOPICS:

- PACKAGES WITH PRIVATE PARTS
    - FULL TYPE DECLARATIONS
- OPERATIONS AVAILABLE ON A PRIVATE TYPE
    - WITHIN PACKAGE
    - OUTSIDE OF PACKAGE
- PRIVATE TYPES WITH DISCRIMINANTS
- RESTRICTIONS
- DEFERRED CONSTANTS

INSTRUCTOR NOTES

- BULLET 2 - WARN THE INSTRUCTORS IN TRAINING THAT L305 STUDENTS MIGHT HAVE A LITTLE TROUBLE WITH THIS. THEY SHOULD EMPHASIZE THAT THE DISCRIMINANT

  - IS PART OF THE DEFINITION OF A VARYING LENGTH STRING OBJECT

  - SHOULD BE VIEWED AS AN ATTRIBUTE OF VARYING LENGTH STRING OBJECT

  - EXAMPLE

    ```
    type Varying_String_Type (Max_Length : Natural) is private;
    ```

    USER DECLARES STRING

    ```
    S : Varying_String_Type (40);
    ```

    GIVING UPPER BOUND ON STRING LENGTH. THE UPPER BOUND IS PART OF THE ABSTRACTION FOR BOUNDED VARYING LENGTH STRINGS

SPECIAL CONSIDERATIONS:

- USING List_Type, EMPHASIZE THAT
    - USER ONLY INTERESTED IN "PUBLIC PART"
    - REPRESENTATION MAY BE CHANGED WITHOUT INVALIDATING LOGIC OF PROGRAMS USING THE ABSTRACTION
        - CHANGE FROM LINEAR LIST TO LINKED LIST
        - OPERATIONS ARE LIST *OPERATIONS*, NOT ARRAY OR POINTER OPERATIONS
- EMPHASIZE THAT NORMALLY REPRESENTATION IS NOT ACCESSIBLE
    - SOMETIMES FULL TYPE MUST BE UNCONSTRAINED RECORD TYPE WITH DISCRIMINANTS FOR PROPER REPRESENTATION OF ABSTRACTION
    - IN SUCH CASES DISCRIMINANTS MUST BE VISIBLE TO USER OF ABSTRACTION
    - ALTHOUGH IMPLEMENTATION DOES SHOW THROUGH, IT IS PART OF THE ABSTRACTION

INSTRUCTOR NOTES

SECTION 8 - LIMITED PRIVATE TYPES

SUMMARY OF MAIN POINTS COVERED:

- INTRODUCE LIMITED PRIVATE TYPES
- DISCUSS WHEN THE USE OF LIMITED PRIVATE TYPES IS APPROPRIATE

MAIN MESSAGES:

- A LIMITED PRIVATE TYPE IS A PRIVATE TYPE THAT HAS NO PREDEFINED ASSIGNMENT (:=), EQUALITY (=), NOR INEQUALITY (/=) OPERATIONS
- LIMITED PRIVATE TYPES ARE ONLY APPROPRIATE WHEN PREDEFINED EQUALITY AND/OR ASSIGNMENT ARE INAPPROPRIATE

SUBTOPICS:

- PROVIDING ASSIGNMENT AND EQUALITY OPERATIONS
- LIMITED COMPOSITE TYPES
  - OPERATIONS
- RESTRICTIONS ON USE
  - FORMAL PARAMETERS
  - NO INITIALIZATION
  - NO DEFAULT EXPRESSIONS
- DEFERRED CONSTANTS

INSTRUCTOR NOTES

- BULLET 2 - THE PROBLEMS OCCUR BECAUSE
  - ITEM 1
    - COMPARING ACCESS VALUES NOT VALUES IN LIST
    - COMPARING UNUSED POSITIONS IN ARRAY
  - ITEM 2
    - COPYING POINTER, NOT LIST; FOR A:=B, IF MODIFY LIST A(B) THEN ALSO MODIFYING LIST B(A)
    - COPYING UNUSED VALUES
- BULLET 3
  - THIS IS AN IMPORTANT POINT THAT L305 INSTRUCTORS MUST MAKE
  - MAKE SURE INSTRUCTORS IN TRAINING UNDERSTAND IT

SPECIAL CONSIDERATIONS:

- EMPHASIZE THAT LIMITED PRIVATE TYPES ARE ONLY APPROPRIATE WHEN PREDEFINED
  - EQUALITY (=), OR
  - ASSIGNMENT (:=)

  ARE INAPPROPRIATE
- EXAMPLES OF List_Type IMPLEMENTATIONS SHOW
  - PREDEFINED EQUALITY (=)
    - GIVING INCORRECT RESULT FOR LINKED LIST VERSION
    - GIVING INCORRECT RESULT FOR ARRAY IMPLEMENTATION
  - PREDEFINED ASSIGNMENT (:=)
    - GIVING INCORRECT RESULT FOR LINKED LIST VERSION
    - INEFFICIENT SOLUTION FOR ARRAY IMPLEMENTATION
- MAKE CLASS AWARE THAT IF LIMITED PRIVATE TYPE SEEMS NECESSARY
  - CONSIDER CHANGING REPRESENTATION SO THAT EQUALITY (=) AND ASSIGNMENT (:=) WORK PROPERLY
  - EXAMPLE : ARRAY IMPLEMENTATION OF List_Type
    - MODIFY List_Type TO BE POINTER TO ARRAY
    - OPERATIONS THAT ALTER A LIST SHOULD YIELD NEW LIST, NOT MODIFY OLD ONE
    - ASSIGNMENT IS NOW ASSIGNMENT OF ACCESS VALUE
    - EQUALITY NOW HOLDS EXACTLY WHEN POINTING TO SAME PHYSICAL COPY
    - NO LONGER NEED LIMITED PRIVATE TYPE

INSTRUCTOR NOTES

- AGAIN URGE THE INSTRUCTORS IN TRAINING TO READ THE CASE STUDIES

SECTION 9 - DESIGNING WITH EXCEPTIONS

SUMMARY OF MAIN POINTS COVERED:

- PROPER USE OF EXCEPTIONS IN PROGRAM DESIGN

MAIN MESSAGES:

- EXCEPTIONS THAT WILL BE VISIBLE BETWEEN MODULES MUST BE DEFINED BY SYSTEM DESIGNERS AND MUST BE USED BY PROGRAMMERS AS SPELLED OUT IN DESIGN
- DESIGNERS AND PROGRAMMERS MUST BOTH DO THEIR JOBS OR USE OF EXCEPTIONS WILL NOT WORK PROPERLY

SUBTOPICS:

- PROPAGATING
- HANDLING
- RE-RAISING

SPECIAL CONSIDERATIONS:

- EMPHASIZE EXCEPTIONS FOR
  - BUILDING FIREWALLS
  - IDIOT PROOFING REUSABLE SOFTWARE COMPONENTS
- Ada CASE STUDIES II GOOD SOURCE FOR DISCUSSION OF PROPER USE OF EXCEPTIONS

INSTRUCTOR NOTES

PART III EXERCISES

- EXERCISE 3 - INTEGER LIST PACKAGE

  - STUDENTS REQUIRED TO WRITE A PACKAGE PROVIDING INTEGER LIST CAPABILITY

    - LIMITED PRIVATE TYPE

    - PROVIDE EXCEPTIONS

  - STUDENTS SHOULD USE SINGLY_LINKED LISTS

  - EXERCISE 8 EXTENDS THIS PROBLEM, SO STUDENTS SHOULD MAKE SURE THEY KEEP THE SOLUTION TO EXERCISE 3

  - MAY BE ASSIGNED AFTER SECTION 9

WARNING: DO NOT ASSIGN THIS EXERCISE IF EXERCISE 2 IS ASSIGNED

INSTRUCTOR NOTES

- GIVE THE INSTRUCTORS IN TRAINING AN OVERVIEW OF WHAT IS COVERED IN PART IV

PART IV - OTHER ABSTRACTION FEATURES

FOCUS OF UNIT:

- DISCUSS ADDITIONAL WAYS TO PROVIDE AN ABSTRACTION
  - USE OF OVERLOADING FOR MORE NATURAL ABSTRACTION
  - USE OF GENERICS FOR GENERAL PURPOSE SOLUTIONS
- USE OF DERIVED TYPES
- USE OF UNCHECKED DEALLOCATION

SECTIONS:

10. OVERLOADING (2:30)
11. GENERICS (3:00)
12. DERIVED TYPES (:50)
13. UNCHECKED DEALLOCATION (:20)

INSTRUCTOR NOTES

- L305 INSTRUCTORS SHOULD EMPHASIZE THE ADDED READABILITY OBTAINED BY USING NAMED ASSOCIATIONS AND SUBPROGRAM OVERLOADING

- MAKE SURE THE INSTRUCTORS IN TRAINING UNDERSTAND THAT IT IS IMPORTANT TO MAKE L305 STUDENTS COMFORTABLE WITH OVERLOADING

  - GOOD PROGRAMMING STYLE WILL MINIMIZE COMPLICATIONS WITH OVERLOADING

- MAIN MESSAGES, BULLET 4: EXAMPLES ARE:

  - "/" FOR RATIONAL NUMBERS
  - "&" VARYING LENGTH STRINGS
  - "+", "*" FOR SETS, VECTORS, MATRICES

SECTION 10 - OVERLOADING

SUMMARY OF MAIN POINTS COVERED:

- INTRODUCE OVERLOADING OF
  - ENUMERATION LITERALS
  - SUBPROGRAMS
  - OPERATOR SYMBOLS
- OVERLOADING RESOLUTION

MAIN MESSAGES:

- OVERLOADING AN ENUMERATION LITERAL IS DESIRABLE WHEN THE LITERAL HAS AN OBVIOUS MEANING IN MORE THAN ONE ENUMERATION TYPE
- SUBPROGRAM OVERLOADING INCREASES PROGRAM READABILITY
- SUBPROGRAM OVERLOADING ELIMINATES NEED TO THINK UP DISTINCT NAMES WHEN DISTINCT NAMES ARE NOT CALLED FOR
- OPERATOR OVERLOADING CAN HELP CREATE <u>PROPER</u> ABSTRACTION

SUBTOPICS:

- OVERLOADING OF ENUMERATION LITERALS
  - CHARACTER LITERALS
- OVERLOADING RESOLUTION
  - AMBIGUITY
  - EXPANDED NAMES
- OVERLOADING OF SUBPROGRAMS
  - PARAMETER AND RESULT TYPE PROFILES
- RULES AND RESTRICTIONS
- OVERLOADING OF OPERATORS

INSTRUCTOR NOTES

BULLET #1

- OVERLOAD RESOLUTION CAN SEEM FORMIDABLE, SO L305 INSTRUCTORS MUST STRESS HOW NATURAL IT REALLY IS
  - COMMON SENSE USUALLY WILL PREVENT PROBLEMS
  - TYPE CONVERSIONS CAN BE USED IN FEW CASES WHERE PROBLEMS OCCUR

BULLET #2

- COMMON EXAMPLES ARE:
  - Put
  - Get
  - Open
  - Close

- L305 INSTRUCTORS SHOULD MAKE IT CLEAR THAT OPERATOR SYMBOLS SHOULD BE OVERLOADED ONLY WHEN IT MAKES SENSE FOR AN ABSTRACTION

SPECIAL CONSIDERATIONS:

- EMPHASIZE THAT RULES FOR OVERLOAD RESOLUTION MAY SEEM COMPLICATED BUT
  - USUALLY CONTEXT ENOUGH TO RESOLVE OVERLOADING
  - NAMED ASSOCIATIONS MAKES PROGRAMS MORE READABLE WHEN USED FORMAL PARAMETER NAMES ARE USED IN OVERLOADING RESOLUTION
  - PLACES WHERE COMPILER CAN'T RESOLVE OVERLOADING AMBIGUITY WOULD GIVE READER OF PROGRAM DIFFICULTY ALSO
  - EXPLICITLY RESOLVING OVERLOADING SELDOM NEEDED BUT EASY TO SPECIFY
- EMPHASIZE THAT OVERLOADING OF SUBPROGRAMS IS COMMONPLACE
- STUDENTS MUST UNDERSTAND THAT OVERLOADING AN OPERATOR IS LIKE OVERLOADING A FUNCTION SUBPROGRAM EXCEPT
  - NORMALLY USE OPERATOR/OPERAND NOTATION RATHER THAN SUBPROGRAM CALL
  - DEFAULT VALUES NOT ALLOWED
  - FOR EQUALITY (=)
    - OVERLOADS INEQUALITY (/=) ALSO
    - BOTH PARAMETERS MUST BE OF SAME LIMITED TYPE
    - RESULT TYPE MUST BE BOOLEAN
- THE RESTRICTIONS ON EQUALITY (=) MAY CAUSE SOME STUDENTS TO THINK
  - PARAMETERS OF OTHER BINARY OPERATORS MUST BE THE SAME
  - OVERLOADED VERSIONS OF OPERATORS WITH BOOLEAN RESULT TYPES MUST HAVE BOOLEAN RESULT TYPES

  EMPHASIZE THAT THIS IS NOT TRUE
- MAKE SURE STUDENTS UNDERSTAND
  - CANNOT DEFINE OWN OPERATOR SYMBOLS
  - ONLY PREDEFINED OPERATORS CAN BE OVERLOADED
    - EXCEPTIONS: in, not in, and then, or else

INSTRUCTOR NOTES

- MAIN MESSAGE BULLET #3

  - L305 INSTRUCTORS SHOULD MAKE SURE L305 CLASS REALIZES THAT GENERIC SUBPROGRAM DECLARATION AND BODY CANNOT BE COMBINED

- CAN BE SEPARATELY COMPILED

- ONLY DECLARATION NEED BE COMPILED BEFORE INSTANTIATION

SECTION 11 - GENERICS

SUMMARY OF MAIN POINTS COVERED:

- INTRODUCE GENERIC UNITS AND INSTANTIATION
- GENERIC FORMAL PARAMETERS
- ILLUSTRATION OF GENERIC PROGRAMMING - CASE STUDY OF GENERALIZATION OF INSERTION SORT

MAIN MESSAGES:

- GENERIC UNITS ELIMINATE REDUNDANT PROGRAMMING
- A GENERIC UNIT IS A PROGRAM UNIT
- GENERIC FORMAL VARIABLES ARE ALLOWED BUT GENERALLY ARE NOT GOOD PROGRAMMING PRACTICE
- GENERIC FORMAL TYPES AND FORMAL SUBPROGRAMS ALLOW FOR CONTROL OVER INSTANTIATION

SUBTOPICS:

- INSTANTIATION
- GENERIC UNITS
  - FORM
  - EXAMPLES
- GENERIC FORMAL PARAMETERS
  - OBJECTS
    - CONSTANTS
    - VARIABLES
  - SUBPROGRAMS
  - TYPES
    - OPERATIONS AVAILABLE
  - ACTUAL PARAMETER SUBSTITUTION
  - DEFAULTS

INSTRUCTOR NOTES

- SPECIAL CONSIDERATIONS, BULLET #3
  - SOME PEOPLE ERRONEOUSLY THINK GENERIC UNITS ARE NOT PRACTICAL BECAUSE
    - COPY OF CODE GENERATED FOR EACH INSTANTIATION, OR
    - RUNTIME DESCRIPTION USED FOR EACH INSTANTIATION
  - THIS PAPER DESCRIBES WHAT A REASONABLE IMPLEMENTATION MIGHT DO

- SPECIAL CONSIDERATIONS, BULLET #5:
  - MAKE SURE THE INSTRUCTORS IN TRAINING UNDERSTAND THAT L305 STUDENTS NEED TO UNDERSTAND ENOUGH OF ALGORITHM TO APPRECIATE THE GENERALIZATIONS. THE L305 STUDENTS DO NOT NEED TO UNDERSTAND EXACTLY HOW THE SORT WORKS.

- SPECIAL CONSIDERATIONS, LAST BULLET:
  - IT IS IMPORTANT FOR AN L305 INSTRUCTOR TO EMPHASIZE REDUCTION OF REDUNDANT PROGRAMMING

SUBTOPICS: (Continued)

- CASE STUDY IN GENERALIZATION - INSERTION SORT
  - CONSIDERATIONS
  - POTENTIAL PROBLEMS

SPECIAL CONSIDERATIONS:

- EMPHASIZE GENERIC PROGRAMMING AS CREATING TEMPLATES
  - INSTANTIATION "FILLS IN THE BLANKS"
- EMPHASIZE "NATURAL" PLACEMENT OF GENERIC UNIT/INSTANTIATION
  - INSTANTIATION ALLOWED WHERE NON-GENERIC VERSION ALLOWED
  - DECLARATION ALLOWED WHERE SUBPROGRAM/PACKAGE DECLARATION ALLOWED
  - BODY ALLOWED WHERE SUBPROGRAM/PACKAGE BODY ALLOWED
- Implementation Implications of Ada Generics IS A GOOD ARTICLE FOR L305 INSTRUCTORS
  - DO NOT EXPLAIN GENERIC INSTANTIATION AS MACRO EXPANSION
  - ARTICLE EXPLAINS HOW A REASONABLE IMPLEMENTATION MIGHT IMPLEMENT GENERIC INSTANTIATION
- MAKE SURE CLASS UNDERSTANDS DISTINCTION BETWEEN GENERIC FORMAL VARIABLES AND GENERIC FORMAL CONSTANTS
- INSERTION SORT EXAMPLE
  - RUN THROUGH THE ALGORITHM TO MAKE SURE CLASS UNDERSTANDS IT
  - USES FIXED LENGTH ARRAY WITH INTEGER INDEX AND COMPONENT
  - FIVE GENERALIZATIONS PRESENTED
- EMPHASIZE THAT GENERIC PROGRAMMING
  - REDUCES REDUNDANT PROGRAMMING
  - REQUIRES CARE WHEN GENERALIZING

INSTRUCTOR NOTES

- THIS SECTION DESCRIBES ONE AREA OF Ada THAT IS A LITTLE BIT PECULIAR
    - AN L305 INSTRUCTOR MUST HAVE A VERY GOOD UNDERSTANDING OF DERIVED TYPES BEFORE ATTEMPTING TO TEACH THIS MATERIAL
    - THE CITED REFERENCE SHOULD BE CONSIDERED MUST READING
- MAIN MESSAGE, BULLET #2:
    - SUBTLE PROBLEMS MAY BE INTRODUCED BY MULTIPLE VERSIONS OF OPERATORS OR BY PRIVATE TYPES IMPLEMENTED BY GENERIC INSTANTIATION

SECTION 12 - DERIVED TYPES

SUMMARY OF MAIN POINTS COVERED:

- DISCUSS USES OF DERIVED TYPES

MAIN MESSAGES:

- A DERIVED TYPE IS A "COPY" OF THE PARENT TYPE
  - ALL PREDEFINED OPERATIONS OF PARENT TYPE ARE AVAILABLE
  - FOR EACH VALUE IN PARENT TYPE THERE IS AN IDENTICAL VALUE IN DERIVED
- DERIVED TYPES ARE USED TO SOLVE SUBTLE PROBLEMS

SUBTOPICS:

- VALUES OF A DERIVED TYPE
- OPERATIONS
- DERIVED SUBPROGRAMS
- USES OF DERIVED TYPES
  - MULTIPLE ABSTRACTIONS
  - MULTIPLE VERSION OF OPERATORS
  - PRIVATE TYPES IMPLEMENTED BY GENERIC INSTANTIATION
  - MULTIPLE REPRESENTATIONS

SPECIAL CONSIDERATIONS:

- NOTE THAT REPRESENTATION CLAUSES WILL BE DISCUSSED IN SECTION 21
- Four Uses For Derived Types, and a Complication GOOD DISCUSSION OF DERIVED TYPES FOR INSTRUCTORS

INSTRUCTOR NOTES

- EACH Ada ACCESS TYPE HAS A STORAGE AREA ASSOCIATED WITH IT. THIS LOCALIZES ALLOCATED VARIABLES FOR A GIVEN ACCESS TYPE. WHEN THE FRAME CONTAINING THE ACCESS TYPE COMPLETES, STORAGE AREA IS FREED. THIS MINIMIZES THE NEED FOR UNCHECKED DEALLOCATION.

SECTION 13 - UNCHECKED DEALLOCATION

SUMMARY OF MAIN POINTS COVERED:

- INTRODUCE THE GENERIC PROCEDURE Unchecked_Deallocation
- GUIDELINES FOR DEALLOCATING VARIABLES

MAIN MESSAGES:

- UNCHECKED DEALLOCATION IS NOT NORMALLY NEEDED AND SHOULD GENERALLY BE AVOIDED

SUBTOPICS:

- DANGERS OF DEALLOCATION

SPECIAL CONSIDERATIONS:

- STUDENTS SHOULD UNDERSTAND THAT
    - UNCHECKED DEALLOCATION SHOULD GENERALLY BE AVOIDED
    - POTENTIALLY DANGEROUS
        - MIGHT TRY TO ACCESS DEALLOCATED VARIABLE
        - CANNOT HAPPEN WITHOUT UNCHECKED DEALLOCATION
    - USE ONLY WHEN NEED IS CLEARLY ESTABLISHED

INSTRUCTOR NOTES

PART IV EXERCISES

- EXERCISES 4 AND 5 ARE ALTERNATIVES TO EACH OTHER
  - BOTH REQUIRE PRIVATE TYPE TO BE PROVIDED
  - BOTH REQUIRE OVERLOADING OF OPERATOR SYMBOLS
  - EXERCISE 4
    - STUDENTS WRITE ARITHMETIC OPERATIONS FOR COMPLEX NUMBERS
    - REVIEW OF COMPLEX NUMBERS PROVIDED BUT STUDENTS WILL BE MORE COMFORTABLE WITH THIS PROBLEM IF PREVIOUSLY EXPOSED (AT HIGH SCHOOL LEVEL) TO COMPLEX NUMBERS
  - EXERCISE 5
    - STUDENTS WRITE VECTOR ARITHMETIC OPERATIONS ON N-DIMENSIONAL VECTORS
    - AGAIN, PREVIOUS EXPOSURE DESIRABLE
    - ONLY PROBLEM REQUIRING DISCRIMINANT WITH PRIVATE TYPE
  - ASSIGN AFTER SECTION 10 ON OVERLOADING

- EXERCISES 6 AND 7 ARE FOLLOW-UPS TO EXERCISES 4 AND 5, RESPECTIVELY
  - GENERALIZE PREVIOUS EXERCISES BY MAKING THEM GENERIC
  - TRANSITION FROM EXERCISE 4 TO EXERCISE 6 INVOLVES TRIVIAL TEXT EDITING
  - TRANSITION FROM EXERCISE 5 TO EXERCISE 7 REQUIRES MORE THOUGHT
    - IN EXERCISE 5, DIMENSIONS SPECIFIED BY DISCRIMINANTS
    - IN EXERCISE 7, DIMENSIONS SPECIFIED BY GENERIC FORMAL CONSTANT

INSTRUCTOR NOTES

- GIVE THE INSTRUCTORS IN TRAINING AN OVERVIEW OF WHAT IS COVERED IN PART V

PART V - APPLICATIONS

FOCUS OF UNIT:

- PRESENT COMMON APPLICATIONS OF DATA STRUCTURES
- DISCUSSION OF VARIOUS SEARCHING AND SORTING ALGORITHMS
- EXPAND ON EARLIER DISCUSSION OF SETS
- INTRODUCE GRAPHS

SECTIONS:

14. GENERIC STACKS (:50)
15. TREES (1:25)
16. SEARCHING (1:45)
17. SORTING (:5C)
18. LINKED LIST IMPLEMENTATION OF SETS (:45)
19. MERGEABLE SETS
20. GRAPHS

INSTRUCTOR NOTES

SECTION 14 - GENERIC STACKS

SUMMARY OF MAIN POINTS COVERED:

- TWO IMPLEMENTATIONS OF STACK TYPES USING GENERIC PACKAGES

MAIN MESSAGES:

- THE FIRST VERSION IS LIKELY TO REQUIRE LESS SPACE AND MAY ALSO BE FASTER
- THE SECOND VERSION USES MORE STORAGE, BUT ALLOWS IT TO BE USED MORE FLEXIBLY

SUBTOPICS:

- BOUNDED VS. UNBOUNDED
- EXAMPLES OF LIMITED PRIVATE TYPE PROVIDED BY GENERIC PACKAGE
- EXAMPLE OF SINGLE-OBJECT GENERIC PACKAGE
- IMPLEMENTATION OF STACK OPERATIONS
  - Push
  - Pop

SPECIAL CONSIDERATIONS:

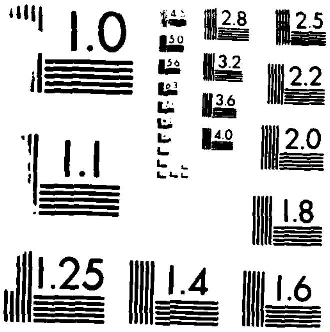
- MAKE SURE CLASS UNDERSTANDS WHY limited private TYPE USED
- EMPHASIZE SINGLE DATA ABSTRACTION CAN BE IMPLEMENTED MANY WAYS
  - IMPLEMENTATION MAY SHOW THROUGH SOMETIMES
  - GOOD DESIGN STRIVES TO MINIMIZE THIS
  - SOMETIMES CANNOT BE PREVENTED
- EMPHASIZE USE OF SUBPROGRAMS TO DETERMINE IF Push OR Pop WILL RAISE AN EXCEPTION
  - Test Stack_Underflow with Is_Empty
  - Test Stack_Overflow with Is_Full, Stack_Space_Available

INSTRUCTOR NOTES

- WARN THE INSTRUCTORS IN TRAINING NOT TO MOVE RAPIDLY THROUGH THE DISCUSSION OF THE HUFFMAN ENCODING. HOWEVER, WHEN PRESENTING CODING DETAILS THEY SHOULD PICK UP THE PACE.

AD-A165 346 ADA (TRADEMARK) TRAINING CURRICULUM INSTRUCTOR'S COURSE 2/3
S500 ADVANCED LANGUAGE MODULES(U) SOFTECH INC WALTHAM
MA 1986 DAAB07-85-C-K506

UNCLASSIFIED F/G 5/9 NL



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS 1963 A

SECTION 15 - TREES

SUMMARY OF MAIN POINTS COVERED:

- INTRODUCE TREES AS A COMMON DATA ABSTRACTION
- THREE REPRESENTATIONS OF BINARY TREES
  - HUFFMAN TREE EXAMPLE
- TWO REPRESENTATIONS OF NON-BINARY TREES

MAIN MESSAGES:

- TREES ARE A COMMON DATA ABSTRACTION
  - EXAMPLE OF A RECURSIVE DATA TYPE

SUBTOPICS:

- BINARY TREES
  - REPRESENTATIONS
  - APPLICATION : HUFFMAN CODES
- NON-BINARY TREES
  - LINEAR LIST REPRESENTATION
  - LINKED LIST REPRESENTATION

SPECIAL CONSIDERATIONS:

- STUDENTS NEED TO LEARN
  - BASIC TERMINOLOGY
  - REPRESENTATIONS
  - FOR BINARY TREES : TRAVERSAL, BUILDING OF TREE
- DO NOT GO FAST THROUGH DESCRIPTION OF HUFFMAN ENCODING
- KEEP IN MIND THAT THE NON-BINARY TREE DISCUSSION IS AN OVERVIEW ONLY
  - MAINLY PICTORIAL
  - ALGORITHMS NOT GIVEN

INSTRUCTOR NOTES

- THE INSTRUCTORS IN TRAINING SHOULD REALIZE THEY ARE NOT TEACHING A COURSE ON SEARCHING

- MAKE SURE THE INSTRUCTORS IN TRAINING UNDERSTAND THEY ARE TO
  - GIVE L305 STUDENTS AN OVERVIEW OF SEARCHING
  - SHOW L305 STUDENTS HOW GENERIC SEARCHING ALGORITHMS CAN BE IMPLEMENTED
  - MAKE L305 STUDENTS AWARE THAT CHOICE OF DATA STRUCTURE AND/OR ALGORITHM CAN HAVE A MAJOR IMPACT ON EFFICIENCY

SECTION 16 - SEARCHING

SUMMARY OF MAIN POINTS COVERED:

- DISCUSSION OF PERFORMANCE
    - WAYS TO COMPARE RELATIVE PERFORMANCE
- PRESENT AND COMPARE VARIOUS SEARCHING ALGORITHMS

MAIN MESSAGES:

- BASIC DATA STRUCTURES ARE USED TO BUILD MORE INTRICATE DATA STRUCTURES
- CHOICE OF GOOD DATA STRUCTURE CAN LEAD TO VERY EFFICIENT ALGORITHM
- DIFFERENT IMPLEMENTATIONS OF SAME ABSTRACTION MAY BE MORE EFFICIENT FOR DIFFERENT APPLICATIONS
- GENERIC UNITS ALLOW GENERAL, TYPE-INDEPENDENT SOLUTIONS

SUBTOPICS:

- PERFORMANCE OF AN ALGORITHM
- LINEAR SEARCH
- BINARY SEARCH
- SEARCH TREES
- HASHING
- PRIORITY QUEUES

SPECIAL CONSIDERATIONS:

- FOR EACH ALGORITHM
    - DISCUSS PERFORMANCE
    - DISCUSS WHEN PARTICULAR SEARCH SHOULD BE USED
- USE PICTURES AS MUCH AS POSSIBLE
    - GIVE OVERVIEW OF CODE
    - POINT OUT INTERESTING FEATURES OF CODE
- KEEP IN MIND THAT THIS IS NOT A COURSE ON SEARCHING

INSTRUCTOR NOTES

- INSTRUCTOR'S NOTES FOR SEARCHING APPLY HERE ALSO

SECTION 17 - SORTING

SUMMARY OF MAIN POINTS COVERED:

- PRESENT AND COMPARE THREE COMMON SORTING ALGORITHMS

MAIN MESSAGES:

- BASIC DATA STRUCTURES ARE USED TO BUILD MORE INTRICATE DATA STRUCTURES
- CHOICE OF GOOD DATA STRUCTURE CAN LEAD TO VERY EFFICIENT ALGORITHM
- DIFFERENT IMPLEMENTATIONS OF SAME ABSTRACTION MAY BE MORE EFFICIENT FOR DIFFERENT APPLICATIONS OR DIFFERENT INPUT DATA
- GENERIC UNITS ALLOW GENERAL, TYPE-INDEPENDENT SOLUTIONS

SUBTOPICS:

- INSERTION SORT
- QUICKSORT
- HEAP SORT

SPECIAL CONSIDERATIONS:

- FOR EACH ALGORITHM
  - DISCUSS PERFORMANCE
  - DISCUSS WHEN PARTICULAR SORT SHOULD BE USED
- USE PICTURES AS MUCH AS POSSIBLE
  - GIVE OVERVIEW OF CODE
  - POINT OUT INTERESTING FEATURES

INSTRUCTOR NOTES

- MAIN MESSAGES, BULLET #1:
  - WITH LINKED LIST VERSION, SET ELEMENTS MAY BE NON-DISCRETE, NON-SCALAR
  - PAYING FOR GENERALITY WITH LESS EFFICIENT IMPLEMENTATION
- MAIN MESSAGES, BULLET #2:
  - NOT REASONABLE TO HAVE COMPLEMENT SINCE LINKED LIST VERSION DOES NOT ASSUME AN ENUMERABLE UNIVERSE OF DISCOURSE

SECTION 18 - LINKED LIST IMPLEMENTATION OF SETS

SUMMARY OF MAIN POINTS COVERED:

- DESCRIBE SETS IMPLEMENTED AS LINKED LISTS RATHER THAN BOOLEAN ARRAYS

MAIN MESSAGES:

- LINKED LIST VERSION GENERALIZES THE BOOLEAN ARRAY VERSION
- LINKED LIST VERSION NOT COMPATIBLE WITH BOOLEAN ARRAY VERSION
- CHOICE DEPENDS ON IMPLEMENTATION

SUBTOPICS:

- REPRESENTATION USING LINKED LISTS
- SET OPERATION
  - COPY-SET
  - EQUALITY
  - UNION
  - INTERSECTION
  - DIFFERENCE
  - SUBSET

SPECIAL CONSIDERATIONS:

- NO CODE IS GIVEN FOR LINKED LIST IMPLEMENTATION, ONLY PICTORIAL ILLUSTRATION
- STUDENTS SHOULD UNDERSTAND THAT WHEN DESIGNING A PACKAGE FOR A DATA ABSTRACTION, CAREFUL CONSIDERATION SHOULD BE GIVEN TO THE OPERATIONS THAT ARE PROVIDED:
  - ANTICIPATE POSSIBLE IMPLEMENTATIONS NEEDED
  - ARE THE OPERATIONS POSSIBLE UNDER THESE IMPLEMENTATIONS?
  - IS IT ACCEPTABLE TO HAVE MULTIPLE VIEWS OF THE SAME ABSTRACTION, E.G., DISCRETE SETS, ARBITRARY SET?
- BOOLEAN ARRAY IMPLEMENTATION IS LEFT AS EXERCISES 15 AND 16

INSTRUCTOR NOTES

- THE MATERIAL IN THIS SECTION IS PROBABLY NEW TO L305 INSTRUCTORS AND L305 STUDENTS
  - L305 INSTRUCTORS MUST HAVE A GOOD UNDERSTANDING OF THE MATERIAL IN THIS SECTION BEFORE TEACHING THE MATERIAL
  - STRONGLY URGE THE INSTRUCTORS IN TRAINING TO CONSULT THE CITED REFERENCE
- SPECIAL CONSIDERATIONS, BULLET #2:
  - THIS IS AN INTERESTING EXAMPLE OF A GENERIC UNIT WITHOUT PARAMETERS. THE PACKAGE IS GENERIC, SO THAT EACH INSTANTIATION WILL PROVIDE A DISTINCT TYPE.

SECTION 19 - MERGEABLE SETS

SUMMARY OF MAIN POINTS COVERED:

- PROVIDE STUDENTS WITH A DIFFERENT VIEW OF SETS, CONCERNED WITH EQUIVALENCE
- GENERIC PACKAGE FOR MERGEABLE SETS

MAIN MESSAGES:

- TWO ELEMENTS ARE EQUIVALENT IF THEY BELONG TO THE SAME SET (EQUIVALENCE CLASS)
- TO MAKE TWO ELEMENTS EQUIVALENT, MERGE THEIR EQUIVALENCE CLASSES

SUBTOPICS:

- OPERATIONS ON MERGEABLE SETS
  - Same_Set
  - Merge_Sets
- TREE IMPLEMENTATION OF MERGEABLE SETS

SPECIAL CONSIDERATIONS:

- THIS VIEW OF SETS MAY BE DIFFICULT FOR SOME STUDENTS
  - USE IS ILLUSTRATED IN SECTION 20 (GRAPHS)
- A GENERIC PACKAGE IS GIVEN FOR MERGEABLE SETS
  - HAS NO GENERIC PARAMETERS. USED ONLY TO CREATE NEW INSTANCE OF THE TYPE PROVIDED BY THE PACKAGE.
  - IMPLEMENTATION IS NOT DIFFICULT
  - EXCELLENT EXAMPLE OF HOW INSIGHT INTO PROBLEM PRODUCES EFFICIENT SOLUTION
    - MEMBERSHIP TEST IS EXPENSIVE IF TREE HEIGHT GETS TOO BIG
    - TREE TRAVERSAL USES INFORMATION GAINED DURING RECURSIVE TRAVERSAL TO REDUCE TREE HEIGHT
  - ENCOURAGE STUDENTS TO CONSULT LITERATURE FOR GOOD IMPLEMENTATION TECHNIQUES
- Data Structures and Algorithms IS A GOOD REFERENCE FOR INSTRUCTORS AND STUDENTS

INSTRUCTOR NOTES

- FOR MANY L305 STUDENTS, THIS MAY BE THEIR FIRST EXPOSURE TO GRAPHS
    - AGAIN ILLUSTRATES RECURRING THEME THAT CHOICE OF A REPRESENTATION DEPENDS ON APPLICATION

SECTION 20 - GRAPHS

SUMMARY OF MAIN POINTS COVERED:

- INTRODUCE GRAPHS AND BASIC DEFINITIONS
- DISCUSS SEVERAL IMPLEMENTATIONS OF GRAPHS
- CONSTRUCTION OF MINIMAL COST SPANNING TREE
    - USE OF DATA ABSTRACTIONS INTRODUCED EARLIER

MAIN MESSAGES:

- GRAPHS ARE A COMMON DATA STRUCTURE
- THE OPERATIONS TO BE PERFORMED MOST FREQUENTLY DETERMINE THE CORRECT CHOICE OF REPRESENTATION

SUBTOPICS:

- BASIC DEFINITIONS
- DIRECTED GRAPHS
- UNDIRECTED GRAPHS
- WEIGHTED GRAPHS
- IMPLEMENTATIONS
    - ADJACENCY MATRIX
    - SUCCESSOR LISTS
    - EDGE SETS
- CONNECTED GRAPHS
- SPANNING TREES
    - CONSTRUCTION OF MINIMAL COST SPANNING TREE

SPECIAL CONSIDERATIONS:

- STUDENTS SHOULD LEARN BASIC DEFINITIONS AND UNDERSTAND REPRESENTATIONS
- EMPHASIZE THAT CORRECT CHOICE OF REPRESENTATION DEPENDS ON OPERATIONS NEEDED MOST
- KEEP IN MIND THAT THIS IS AN OVERVIEW TO MAKE L305 STUDENTS AWARE OF ANOTHER COMMON DATA STRUCTURE

INSTRUCTOR NOTES

PART V - EXERCISES

- EXERCISE 8 OR 9
  - CAN BE ASSIGNED AFTER SECTION 14 - GENERIC STACKS
  - EXERCISE 8 REQUIRES THE STUDENT TO WRITE A GENERIC LIST PACKAGE
    - REQUIRES SOLID UNDERSTANDING OF GENERIC UNITS
    - MOST OF WORK DONE IN EXERCISE 3 (INTEGER LISTS)
    - PACKAGE USED IN REST OF COURSE AND ASSUMED FOR EXERCISES 10, 11, AND 14
    - IF NOT ASSIGNED THEN REVIEW PACKAGE SPECIFICATION IN CLASS

- EXERCISE 9 REQUIRES STUDENTS TO WRITE GENERIC QUEUE PACKAGE
  - ASSUMES SOLUTION TO 8 IS NOT USED

- EXERCISE 10 OR 11
  - CAN BE ASSIGNED AFTER SECTION 14 - GENERIC STACKS
  - BOTH SHOW STUDENTS HOW GENERAL PURPOSE LIST PACKAGE FROM EXERCISE 8 CAN BE USED TO IMPLEMENT HIGHER-LEVEL DATA ABSTRACTION
  - BOTH SOLUTIONS INTRICATE AND TIME CONSUMING
  - BOTH SOLUTIONS FOLLOW ALMOST THE SAME LOGIC
  - EXERCISE 10 REQUIRES STUDENTS TO WRITE PACKAGE FOR UNBOUNDED NATURAL NUMBERS
    - ADDITION AND MULTIPLICATION (FOR AMBITIOUS STUDENTS ONLY)
    - REQUIRES CARRY
  - EXERCISE 11 REQUIRES STUDENTS TO WRITE GENERIC POLYNOMIAL PACKAGE
    - ADDITION AND MULTIPLICATION (FOR AMBITIOUS STUDENTS ONLY)
    - CARRY NOT NEEDED, BUT POLYNOMIALS ADDITION/MULTIPLICATION LESS FAMILIAR

- EXERCISE 12 OR 13
  - CAN BE ASSIGNED AFTER SECTION 15 - TREES
  - SIMPLE EXAMPLES IN TREE MANIPULATION AND RECURSION
  - SHOULD ASSIGN ONE OF THESE PROBLEMS
  - EXERCISE 12 REVERSES A BINARY TREE
  - EXERCISE 13 SUMS THE LEAVES OF A TREE

INSTRUCTOR NOTES

PART V - EXERCISES - continued

- EXERCISE 14
  - CAN BE ASSIGNED AFTER SECTION 16 - SORTING
  - FAIRLY SHORT EXERCISE
  - MODIFICATION OF A PRIORITY QUEUE PACKAGE DEVELOPED IN CLASS
  - DEMONSTRATES THERE CAN BE MANY IMPLEMENTATIONS OF THE SAME DATA ABSTRACTION, WITH DIFFERENT PERFORMANCE CHARACTERISTICS
  - PROVIDES EXPERIENCE IN USE OF A PREVIOUSLY WRITTEN GENERIC PACKAGE
  - FORCES STUDENTS TO CONFRONT SOME OF THE NAMING PROBLEMS THAT CAN ARISE WHEN USING DERIVED TYPES

- EXERCISE 15 AND 16
  - CAN BE ASSIGNED AFTER SECTION 18 - LINKED LIST IMPLEMENTATION OF SETS
  - STRONGLY RECOMMENDED
  - EXERCISE 15 REQUIRES STUDENTS TO IMPLEMENT A GENERIC SET PACKAGE USING BOOLEAN ARRAYS
  - FAIRLY SIMPLE
  - PROVIDES REVIEW OF ESSENTIAL CONCEPTS PRESENTED IN COURSE
    - PACKAGES
    - PRIVATE TYPES
    - GENERICS
    - OVERLOADING
  - GOOD AS FINAL EXERCISE(S)
  - SHOULD HELP Pascal PROGRAMMERS FEEL MORE COMFORTABLE WITH Ada BY SHOWING THEM A CONVENIENT WAY TO OBTAIN THE EQUIVALENT OF A Pascal SET TYPE
  - EXERCISE 16 EXTENDS EXERCISE 15
    - INTENDED FOR THOSE WHO FINISH EXERCISE 15 EARLY, OR
    - FOR THOSE WHO DID NOT FIND EXERCISE 15 SUFFICIENTLY CHALLENGING

INSTRUCTOR NOTES

- GIVE THE INSTRUCTORS IN TRAINING AN OVERVIEW OF PART VI OF L305

PART VI - LOW-LEVEL AND IMPLEMENTATION-DEPENDENT PROGRAMMING

FOCUS OF UNIT:

- REASONS FOR USING LOW-LEVEL FEATURES
- EFFECT ON PORTABILITY
- HOW TO ENCAPSULATE MACHINE DEPENDENCIES

SECTIONS:

21. LOW-LEVEL AND IMPLEMENTATION-DEPENDENT FEATURES (3:00)
22. EXAMPLE OF LOW-LEVEL PROGRAMMING (1:00)

INSTRUCTOR NOTES

SECTION 21 - LOW-LEVEL AND IMPLEMENTATION-DEPENDENT FEATURES

SUMMARY OF MAIN POINTS COVERED:

- CAPABILITIES PROVIDED BY LOW-LEVEL FEATURES, WITH GUIDELINES FOR USE
- DISCUSS IMPLEMENTATION-DEPENDENT FEATURES AND THEIR EFFECT ON PORTABILITY

MAIN MESSAGES:

- Ada PROVIDES A WIDE RANGE OF FEATURES FOR INTERFACING WITH
  - HARDWARE
  - EXISTING SOFTWARE
- SOME OF THE LOW-LEVEL FEATURES NEED NOT BE PROVIDED BY ANY IMPLEMENTATION
- CONSIDER THE GUIDELINES SUGGESTED IN THIS SECTION WHEN DECIDING WHETHER OR NOT TO USE LOW-LEVEL FEATURES

SUBTOPICS:

- GUIDELINES FOR USING LOW-LEVEL FEATURES
- EFFECT ON PORTABILITY
- THE PACKAGE SYSTEM
- REPRESENTATION ATTRIBUTES
- PRAGMAS
- UNCHECKED CONVERSION
- INTERFACE WITH OTHER LANGUAGES
- CODE PROCEDURES
- DEVICE LEVEL INPUT/OUTPUT

SPECIAL CONSIDERATIONS:

- L305 INSTRUCTORS SHOULD EMPHASIZE HIGH LEVEL WAY IN WHICH Ada PROVIDES FOR LOW-LEVEL AND IMPLEMENTATION-DEPENDENT FEATURES

INSTRUCTOR NOTES

- MAKE SURE THE INSTRUCTORS IN TRAINING UNDERSTAND THAT IN ADDITION TO ILLUSTRATING THE FEATURES OF THE LAST SECTION, THIS SECTION SHOWS HOW HIGH-LEVEL AND LOW-LEVEL FEATURES CAN BE USED TOGETHER TO PROVIDE DATA ABSTRACTION

  - DETAILS OF EXAMPLE ARE IMPORTANT ONLY IN THAT L305 CLASS SHOULD FIND IT REALISTIC

  - L305 STUDENTS NEED TO REALIZE Ada IS A VIABLE LANGUAGE FOR LOW-LEVEL PROGRAMMING

SECTION 22 - EXAMPLE OF LOW-LEVEL PROGRAMMING

SUMMARY OF MAIN POINTS COVERED:

- PRESENT AN APPLICATION OF LOW-LEVEL PROGRAMMING FEATURES : AN ANTENNA-TUNER-INTERFACE
- ILLUSTRATE DEVELOPMENT OF SOFTWARE INTERFACE GIVEN HARDWARE DESCRIPTION
    - IMPLEMENTATION DEVELOPED NEXT

MAIN MESSAGES:

- REPRESENTATION ATTRIBUTES CAN BE USED FOR MAPPING DATA OBJECTS TO HARDWARE REGISTERS
- Low_Level_IO PACKAGE CAN BE USED TO PROVIDE HIGHER LEVEL OF ABSTRACTION
- UNCHECKED_CONVERSION CAN BE USED TO CONVERT BETWEEN HARDWARE INTERFACE VIEW OF DATA AND Low_Level_IO VIEW OF DATA

SUBTOPICS:

- HARDWARE SPECIFICATION
- SOFTWARE INTERFACE
    - REQUIREMENTS
    - RATIONALE
- IMPLEMENTATIONS

SPECIAL CONSIDERATIONS:

- EMPHASIZE THAT
    - LOW LEVEL PROGRAMMING CAN BE PERFORMED EFFECTIVELY IN Ada
    - ADVANTAGE OBTAINED USING Ada IS ABILITY TO PROVIDE HIGH-LEVEL ABSTRACTION TO USERS OF THE ANTENNA TUNER
- POINT OUT THAT THE VERSION OF Low_Level_IO PRESENTED IS HYPOTHETICAL, AS IS THE HARDWARE SPECIFICATION

INSTRUCTOR NOTES

- STUDENTS REQUIRING MORE IN-DEPTH COVERAGE SHOULD TAKE L401 OR L303

PART VII - REMAINING Ada FEATURES

FOCUS OF UNIT:

- PROVIDE OVERVIEW OF MAIN TASKING FEATURES OF Ada

SECTIONS:

23. OVERVIEW OF Ada TASKING (2:00)

INSTRUCTOR NOTES

- L305 INSTRUCTORS MUST BE WELL-VERSED IN Ada TASKING AT LEAST TO THE L303 LEVEL. RECOMMEND L401 LEVEL.

- L305 STUDENTS SHOULD HAVE A READING KNOWLEDGE OF Ada TASKING UPON COMPLETING THIS SECTION.

SECTION 23 - OVERVIEW OF Ada TASKING

SUMMARY OF MAIN POINTS COVERED:

- OVERVIEW OF TASKING TOPICS

MAIN MESSAGES:

- TASKS ALLOW SEVERAL SEQUENCES OF ACTIONS TO BE PERFORMED SIMULTANEOUSLY

SUBTOPICS:

- Ada VIEW OF TASKING
- DECLARING TASK TYPES AND TASK OBJECTS
- RENDEZVOUS
- ACTIVATION AND TERMINATION
- ENTRY CALLS/ACCEPT STATEMENTS
- SELECTIVE WAITS
- GUARDS
- TASK ATTRIBUTES

SPECIAL CONSIDERATIONS:

- THIS OVERVIEW MAY NOT BE ENOUGH FOR SOME STUDENTS
  - THEY MAY ASK QUESTIONS BEYOND THE SCOPE OF THE COURSE
    - ANSWERING THESE QUESTIONS MAY SIDETRACK YOU
    - STUDENTS NEEDING MORE DETAILS SHOULD TAKE L303 OR L401
  - EXPECT MANY QUESTIONS ABOUT TASKING
    - L305 INSTRUCTORS SHOULD BE WELL-VERSED IN Ada TASKING, AT LEAST TO L303 LEVEL
- REMEMBER THIS IS AN OVERVIEW
  - FOR SOME STUDENTS, THIS MAY BE THE ONLY CLASSROOM EXPOSURE TO Ada TASKING
  - IF L305 IS BEING TAKEN AS A PREREQUISITE FOR L303 OR L401, THIS SECTION CAN BE SKIPPED

INSTRUCTOR NOTES

- ALLOW 145 MINUTES (2 HOURS/25 MINUTES) FOR THIS SECTION

SECTION 3

L401

REAL TIME SYSTEMS IN Ada

INSTRUCTOR NOTES

- GO OVER THIS SLIDE IN ENOUGH DETAIL TO REMIND THE INSTRUCTORS IN TRAINING OF WHAT TOPICS WILL BE COVERED.

- SUGGEST TO THE INSTRUCTORS IN TRAINING THAT THEY DO THE SAME WHEN THEY TEACH L401.

L401 TABLE OF CONTENTS

INSTRUCTOR NOTES

- GIVE THE INSTRUCTORS IN TRAINING AN OVERVIEW OF WHAT IS COVERED IN PART I AND WHAT THE OBJECTIVES ARE

- TARGET TEACHING TIME IS IN PARENTHESES
  - INCLUDED TO GIVE INSTRUCTORS IN TRAINING AN IDEA OF HOW MUCH TIME IS DEVOTED TO EACH TOPIC
  - EMPHASIZE THAT THIS IS TARGET TIME. ACTUAL TIME MAY VARY DEPENDING ON CLASS NEEDS/BACKGROUND.

- EMPHASIZE THAT THIS PART OF THE COURSE SHOULD BE KEPT LIGHT

PART I - CONCURRENT PROGRAMMING CONCEPTS

FOCUS OF UNIT:

- BASIC CONCEPTS OF CONCURRENT PROGRAMMING
- NEEDS AND USES FOR CONCURRENCY
- COMMON PITFALLS IN CONCURRENT PROGRAMMING
- SERVES AS INTRODUCTION TO CONCURRENT PROGRAMMING FOR THOSE WITHOUT CONCURRENT PROGRAMMING BACKGROUND

SECTIONS:

- SECTION 1 - CONCURRENT PROGRAMMING CONCEPTS (1:00)
- SECTION 2 - REASONS FOR CONCURRENCY (1:00)
- SECTION 3 - CONCURRENT PROGRAMMING PROBLEMS (:50)

INSTRUCTOR NOTES

- THIS SECTION IS ENTIRELY INTRODUCTORY.

- MAIN MESSAGES
  - THE CONCEPTS MENTIONED IN THE LAST FOUR BULLETS WILL BE USED THROUGHOUT THE L401 MODULE. THE INSTRUCTORS IN TRAINING MUST BE CERTAIN THAT L401 STUDENTS UNDERSTAND THESE CONCEPTS
  - FOR STUDENTS WHO ARE BEING EXPOSED TO CONCURRENCY FOR THE FIRST TIME, AN L401 INSTRUCTOR MIGHT NEED TO RE-EMPHASIZE THESE CONCEPTS EACH TIME THEY APPEAR AGAIN IN THE COURSE

SECTION 1 - CONCURRENT PROCESSES

SUMMARY OF MAIN POINTS COVERED:

- INTRODUCE BASIC CONCEPTS OF CONCURRENT PROGRAMMING

MAIN MESSAGES:

- A PROCESS IS A SEQUENCE OF ACTIONS PERFORMED IN CARRYING OUT A PROGRAM, SEVERAL OF WHICH CAN BE IN PROGRESS AT THE SAME TIME
- PROCESS IS NOT SYNONYMOUS WITH Ada TASK
- EACH ASYNCHRONOUS PROCESS EXECUTES AT ITS OWN RATE; RELATIVE PROGRESS UNPREDICTABLE
  - MAY NEED TO SYNCHRONIZE OCCASIONALLY
- Ada PROGRAMS ARE REENTRANT
  - EACH PROCESS HAS ITS OWN DATA AREA
  - DIFFERENT PROCESS MAY EXECUTE THE SAME SEQUENCE OF STATEMENTS
  - PROGRAMS DO NOT MODIFY THEMSELVES
- Ada PROGRAM MAY SPECIFY ACTIONS TO BE PERFORMED BY MORE THAN ONE PROCESS
  - EACH PROCESS EXECUTES ACTIONS IN SEQUENCE
  - SEVERAL SEQUENCES MAY BE IN PROGRESS AT ONCE
- RUNTIME SYSTEM PROVIDES A VIRTUAL PROCESSOR FOR EACH PROCESS

SUBTOPICS:

- SINGLE- AND MULTIPLE-PROCESS PROGRAMS
- PROCESSORS
  - VIRTUAL
- OVERLAPPED AND INTERLEAVED CONCURRENCY
- RUNTIME SYSTEMS
- ASYNCHRONOUS PROCESSES
  - SYNCHRONIZATION

INSTRUCTOR NOTES

- FOR SOME STUDENTS THIS MAY BE THE FIRST TIME THEY HAVE DEALT WITH CONCURRENT PROGRAMMING
  - FOR SUCH STUDENTS, THE MACARONI AND CREAM SAUCE EXAMPLE WILL ILLUSTRATE MANY IMPORTANT FEATURES, TELL THE INSTRUCTORS IN TRAINING TO GO THROUGH THE EXAMPLE SLOWLY
  - FOR MORE EXPERIENCED STUDENTS, THE EXAMPLE SHOULD BE PRESENTED AT A FASTER PACE

- MAKE SURE THE INSTRUCTORS IN TRAINING UNDERSTAND THAT A PROCESS IS NOT AN Ada TASK

SPECIAL CONSIDERATIONS:

- USUALLY WE THINK OF A PROGRAM'S EXECUTION AS CONSISTING OF A SINGLE SEQUENCE OF ACTIONS
  - SOME STUDENTS MAY HAVE DIFFICULTY UNDERSTANDING HOW A PROGRAM CAN CONSIST OF MORE THAN ONE SEQUENCE OF ACTIONS
  - MACARONI AND CREAM SAUCE EXAMPLE ILLUSTRATES THE CONCEPTS AND IS USEFUL IN EXPLAINING OVERLAPPED AND INTERLEAVED CONCURRENCY
- DISCUSSION OF CONCURRENT PROCESSES SHOULD BE KEPT ALMOST ENTIRELY LANGUAGE INDEPENDENT
  - PART I DEALS WITH PROCESSES WHICH ARE INDEPENDENT OF ANY PROGRAMMING LANGUAGE
  - PROCESS IS NOT SYNONYMOUS WITH Ada TASK
  - Ada TASK OBJECTS ARE DESCRIBED IN SECTION 4, TASK TYPES AND TASK OBJECTS
  - MAKE SURE YOU USE THE TERM "PROCESS" NOT "TASK" THROUGHOUT PART I.
- RULES OF Ada ALLOW FOR MANY DIFFERENT IMPLEMENTATIONS OF THE RUNTIME SYSTEM
  - MANY STUDENTS WILL BE CONCERNED ABOUT THIS
  - THEY NEED TO UNDERSTAND THAT A PARTICULAR RUNTIME SYSTEM MAY BE GOOD FOR SOME APPLICATIONS AND NOT GOOD FOR OTHERS
    - EVENTUALLY THERE WILL BE MANY OFF-THE-SHELF RUNTIME SYSTEMS
    - Ada RUNTIME SYSTEMS CAN BE ADAPTED TO MEET APPLICATION NEEDS AS LONG AS THEY DON'T VIOLATE Ada RULES
- FOR STUDENTS WITH CONCURRENT PROGRAMMING BACKGROUND
  - YOU CAN MOVE THROUGH THIS SECTION AT A FASTER THAN NORMAL PACE
  - SLOW DOWN FOR RUNTIME SYSTEMS MATERIAL

INSTRUCTOR NOTES

- WHILE THIS SECTION IS ESPECIALLY IMPORTANT FOR STUDENTS BEING EXPOSED TO CONCURRENCY FOR THE FIRST TIME, THE INSTRUCTORS IN TRAINING SHOULD REALIZE THAT SOME OF REASONS DISCUSSED MAY BE NEW TO EVEN EXPERIENCED REAL TIME PROGRAMMERS
    - THIS WILL ALMOST CERTAINLY BE TRUE WHEN DISCUSSING THE FOURTH REASON: LOGICAL DECOMPOSITION OF A COMPLEX PROGRAM

- INSTRUCTORS IN TRAINING SHOULD UNDERSTAND THAT DETAILED DISCUSSIONS OF THE EXAMPLES SHOULD BE AVOIDED. THE EXAMPLES WILL BE COVERED IN DETAIL LATER IN THE COURSE AND, IN SEVERAL CASES, THE Ada CODE WILL BE PRESENTED

SECTION 2 - REASONS FOR CONCURRENCY

SUMMARY OF MAIN POINTS COVERED:

- SKETCH SOME REASONS FOR WRITING CONCURRENT PROGRAMS
  - MANAGEMENT OF SIMULTANEOUS REAL TIME ACTIVITIES
  - SIMULATION OF SIMULTANEOUS ACTIVITIES
  - PARALLEL COMPUTATION
  - LOGICAL DECOMPOSITION OF A COMPLEX PROBLEM
- INTRODUCE CYCLIC EXECUTIVES AND SINGLE-THREAD PROCESSES

MAIN MESSAGES:

- CONCURRENT PROGRAMMING IS A WAY OF THINKING ABOUT THE STRUCTURE OF A PROBLEM, NOT JUST A WAY OF SPECIFYING THAT CERTAIN ACTIONS CAN BE EXECUTED SIMULTANEOUSLY
- CONCURRENT PROGRAMMING CAN BE USED TO SOLVE A "SEQUENTIAL PROBLEM"

SUBTOPICS:

- MANAGEMENT OF SIMULTANEOUS REAL-WORLD ACTIVITIES
  - CYCLIC EXECUTIVES
  - SINGLE-THREAD PROCESSES
- SIMULATING SIMULTANEOUS REAL-WORLD ACTIVITIES
  - RADAR SIMULATION EXAMPLE
- PARALLEL COMPUTATION
  - PARALLEL SORTING EXAMPLE
- LOGICAL DECOMPOSITION OF A COMPLEX PROBLEM
- COMMON THEMES

INSTRUCTOR NOTES

SPECIAL CONSIDERATIONS:

- WHAT IS CALLED AN ACTIVITY IN THIS MODULE IS GENERALLY CALLED A TASK BY REAL TIME PROGRAMMERS
- EMPHASIZE THAT HARD SEQUENTIAL PROBLEMS CAN SOMETIMES BE SOLVED MORE EASILY AS SEQUENCE OF SIMPLER CONCURRENT PROCESSES
- MAY SURPRISE STUDENTS THAT CONCURRENT PROGRAMMING CAN BE USED TO SOLVE A "SEQUENTIAL" PROBLEM
- EMPHASIZE COMMON THEMES
  - SEVERAL CONCEPTUAL THREADS
  - DETAILS OF SCHEDULING AND INTERLEAVING HANDLED BY RUNTIME SYSTEM, AND DO NOT APPEAR IN THE PROGRAM
  - STRUCTURE OF THE PROGRAM REFLECTS CONCEPTUAL THREADS
- AVOID DETAILED DISCUSSIONS OF THE EXAMPLES
  - WILL BE DISCUSSED IN DETAIL LATER IN THE MODULE
  - Ada CODE WILL BE USED

INSTRUCTOR NOTES

- AGAIN, EVEN EXPERIENCED REAL TIME PROGRAMMERS MAY NOT BE FAMILIAR WITH THE NATURE OF THESE PROBLEMS, JUST THE SYMPTOMS
    - INSTRUCTORS IN TRAINING SHOULD MAKE SURE THEY FEEL COMFORTABLE WITH THESE TOPICS

- THE BUG HEARD ROUND THE WORLD IS A GOOD EXAMPLE OF THE PROBLEMS THAT CAN OCCUR IN REAL TIME PROGRAMMING
    - INSTRUCTORS IN TRAINING SHOULD READ THIS ARTICLE AS BACKGROUND BUT THEY DO NOT NEED TO UNDERSTAND ALL OF THE DETAILS

SECTION 3 - CONCURRENT PROGRAMMING PROBLEMS

SUMMARY OF MAIN POINTS COVERED:

- DESCRIBES THE MOST COMMON PITFALLS IN CONCURRENT PROGRAMMING

MAIN MESSAGES:

- CONCURRENT PROGRAMMING IS TRICKY, ENTAILING MANY SUBTLE PROBLEMS NOT FOUND IN SINGLE-PROCESS PROGRAMMING

SUBTOPICS:

- SIMULTANEOUS UPDATE
    - RACE CONDITION
- DEADLOCK
- STARVATION
- PROCESS COOPERATION
    - SYNCHRONIZATION
    - COMMUNICATION

SPECIAL CONSIDERATIONS:

- PROBLEMS DESCRIBED THROUGH
    - PICTURES DEALING WITH SOUTHERN PATHETIC RAILROAD
    - ANECDOTES
        - ADD ANECDOTES YOU ARE AWARE OF
        - ENCOURAGE STUDENTS TO DO THE SAME
- Ada CODE USED FOR THE FIRST TIME IN THIS MODULE
    - TASKING FEATURES NOT USED
    - PREPARES STUDENT FOR REALIZATION THAT INTUITION DEVELOPED THROUGH YEARS OF SINGLE-PROCESS PROGRAMMING IS NOT SUFFICIENT
- THE ANECDOTE ON PROCESS COOPERATION IS CONDENSED FROM THE BUG HEARD ROUND THE WORLD
    - THE PROBLEM DESCRIBED IS INTRICATE
    - DON'T WORRY IF YOU AND THE CLASS DON'T UNDERSTAND ALL THE INTRICACIES
    - POINT IS THAT TASK SYNCHRONIZATION CAN BE COMPLEX, AND THAT TASKS CAN INTERACT IN SURPRISING WAYS THAT ARE DIFFICULT TO UNDERSTAND
- KEEP THIS SECTION LIGHT

INSTRUCTOR NOTES

SECTION 3 - EXERCISE

- THIS EXERCISE ASKS THE STUDENTS TO CONTINUE WITH SOUTHERN PATHETIC'S ATTEMPTS TO RUN A REASONABLE RAILROAD
  - PROBLEM IN PROCESS COOPERATION
  - NOT ALLOWED TO USE Ada TASKING FEATURES
- THIS IS A DIFFICULT PROBLEM TO SOLVE, AND MOST STUDENTS (PROBABLY ALL STUDENTS) WILL NOT BE ABLE TO SOLVE IT
  - THE IDEA IS JUST TO GET THEM THINKING ABOUT THE PROBLEMS INVOLVED
  - SOLUTION DESCRIBES BOTH A CORRECT APPROACH AND SEVERAL NEAR-MISSES REFLECTING COMMON ERRORS

INSTRUCTOR NOTES

- GIVE THE INSTRUCTORS IN TRAINING AN OVERVIEW OF WHAT IS COVERED IN PART II AND WHAT THE OBJECTIVES ARE.

PART II - Ada TASKING CONCEPTS

FOCUS OF UNIT:

- INTRODUCES BASIC Ada TASKING CONCEPTS
- STUDENTS SHOULD UNDERSTAND
  - HOW TO WRITE A TASK DECLARATION
  - WHEN A TASK OBJECT STARTS EXECUTING
  - WHEN A TASK OBJECT ENDS ITS EXECUTION

SECTIONS:

- SECTION 4 - TASK TYPES AND TASK OBJECTS (:20)
- SECTION 5 - TASK DECLARATIONS AND TASK TYPES (1:00/:30)
- SECTION 6 - TASK ACTIVATION AND TERMINATION (:45)

INSTRUCTOR NOTES

- INSTRUCTORS IN TRAINING NEED TO UNDERSTAND THAT L401 STUDENTS SHOULD BE ENCOURAGED TO THINK OF TASK TYPES AS BEING SIMILAR TO OTHER Ada TYPES WITH TASK OBJECTS AS VALUES IN THE TYPE
  - L401 STUDENTS MUST UNDERSTAND THIS VIEW EVENTUALLY TO SUCCEED IN THIS COURSE
  - L401 INSTRUCTORS SHOULD BE SENSITIVE TO THIS NEED, BUT SHOULD REALIZE THAT THE EXAMPLES IN THIS COURSE WILL REINFORCE THIS VIEW

SECTION 4 - TASK TYPES AND TASK OBJECTS

SUMMARY OF MAIN POINTS COVERED:

- INTRODUCE TASK TYPES AND TASK OBJECTS

MAIN MESSAGES:

- A TASK TYPE HAS
  - A SET OF VALUES (CALLED TASK OBJECTS)
  - A SET OF WAYS TO MANIPULATE THESE VALUES
- IN THIS SENSE A TASK TYPE IS JUST LIKE ANY OTHER TYPE

SUBTOPICS:

- REVIEW TYPES
- EXAMPLE OF A TASK TYPE

SPECIAL CONSIDERATIONS:

- THE CONCEPTS OF THIS SECTION WILL BE NEW TO MANY STUDENTS, SO GO THROUGH IT SLOWLY
- IT MIGHT BE DIFFICULT FOR SOME STUDENTS TO ACCEPT TASKS AS BEING SIMILAR TO OTHER Ada TYPES
  - DIFFERS FROM TRADITIONAL VIEW OF PROCESS
  - MAY BE DIFFICULT TO ACCEPT SOMETHING THAT "EXECUTES" AS DATA
  - MAKE SURE CLASS UNDERSTANDS THAT IF THEY DO NOT UNDERSTAND THIS VIEW NOW, THEY WILL LATER IN COURSE AFTER SEEING SEVERAL EXAMPLES
- STARTING WITH THIS SECTION, STOP USING THE TERM PROCESS
  - FOR NOW, TALK ABOUT TASK OBJECTS
  - NEXT SECTION CAN SIMPLIFY TO JUST TASKS

INSTRUCTOR NOTES

SECTION 5 - TASK DECLARATIONS AND TASK TYPES

SUMMARY OF MAIN POINTS COVERED:

- FORM AND PLACEMENT OF TASK DECLARATIONS AND TASK BODIES

MAIN MESSAGES:

- A TASK UNIT HAS A FORM AND ROLE ANALOGOUS TO Ada'S OTHER PROGRAM UNITS
- TASKS SHOULD BE THOUGHT OF AS DATA OBJECTS
- TASK TYPES ARE LIMITED TYPES

SUBTOPICS:

- BASIC TASK TYPE DECLARATION
  - SYNTAX
  - EXAMPLE
- TASK BODY
  - SYNTAX
  - EXAMPLE
- USE OF TASK TYPES
- COMPOSITE TYPES USING TASKS
- TASK UNITS
- ANONYMOUS TASK TYPES

INSTRUCTOR NOTES

SPECIAL CONSIDERATIONS:

- MAKE SURE CLASS UNDERSTANDS THAT SEPARATE COMPILATION OF TASK UNITS AS LIBRARY UNITS IS NOT SUPPORTED, BUT THE EFFECT CAN BE ACHIEVED USING PACKAGES
  - SIMILARLY GENERIC TASK UNITS NOT SUPPORTED BUT THE EFFECT CAN BE ACHIEVED USING GENERIC PACKAGES
- MAKE SURE THE CLASS UNDERSTANDS THAT TASK TYPES MAY APPEAR AS
  - SIMPLE VARIABLES
  - ARRAY OR RECORD COMPONENTS
  - COMPONENTS OF ACCESS TYPE
  - ETC.
- EMPHASIS SHOULD BE ON TASK UNITS DEFINING TASK TYPES
  - OCCASIONALLY NEED ONE-OF-A-KIND TASKS
  - ANALOGY MADE WITH ONE-OF-A-KIND ARRAYS
- A TONGUE-IN-CHEEK EXAMPLE OF A GEIGER COUNTER
  - UNLESS YOU HIT THE CLASS OVER THE HEAD WITH THIS EXAMPLE, THEY WILL NOT NOTICE THIS ATTEMPT AT HUMOR
- REMEMBER, YOU WANT THE CLASS TO THINK OF Ada TASKS AS DATA OBJECTS

INSTRUCTOR NOTES

SECTION 5 - EXERCISE

- THIS EXERCISE TESTS THE STUDENTS' UNDERSTANDING OF THE FORM OF A TASK UNIT
  - MUST PROVIDE TASK DECLARATION WITH TWO ENTRY DECLARATIONS
  - SKELETON TASK BODY CONTAINING DATA DECLARATIONS
- STUDENTS ARE ALSO REQUIRED TO USE THE TASK TYPE IN DEFINING AN ARRAY OF TASKS
- CLASS SHOULD HAVE NO PROBLEMS WITH THIS EXERCISE

INSTRUCTOR NOTES

- THE INSTRUCTORS IN TRAINING SHOULD BE PREPARED FOR POSSIBLE PROBLEMS RESULTING FROM CONFUSION OVER "INITIATION" AND "ACTIVATION"

- SPECIAL CONSIDERATIONS, BULLET #1

  - AS LONG AS A TASK IS EXECUTING, "RESOURCES" IT MAY NEED MAY NOT CEASE TO EXIST
  - MASTER DESCRIBES WHAT THESE "RESOURCES" ARE

SECTION 6 - TASK ACTIVATION AND TERMINATION

SUMMARY OF MAIN POINTS COVERED:

- DESCRIBE WHEN A TASK IS ACTIVATED AND WHEN IT IS TERMINATED
- EXAMPLES OF ACTIVATION AND TERMINATION OF BOTH DECLARED AND ALLOCATED TASKS

MAIN MESSAGES:

- A TASK BEGINS PARALLEL EXECUTION (IS ACTIVATED) ABOUT THE TIME IT IS CREATED
- A TASK HAS TERMINATED IF IT HAS COMPLETED AND ALL DEPENDENT TASKS, IF IT HAS ANY, HAVE TERMINATED

SUBTOPICS:

- DECLARED TASKS
- ALLOCATED TASKS
- TASKS AND (NON-LIBRARY) PACKAGES
- MASTERS AND DEPENDENCY
- COMPLETION

SPECIAL CONSIDERATIONS:

- EMPHASIZE MASTER OF A TASK AND DEPENDENCY AS A NATURAL WAY OF LOOKING AT TERMINATION
- THE DEFINITION OF TERMINATION PRESENTED HERE IS SIMPLIFIED BY NOT INTRODUCING THE TERMINATE NOW. IT IS INTRODUCED LATER IN SECTION 8 WHEN SELECTIVE WAIT STATEMENTS ARE INTRODUCED.
- FOR STUDENTS FAMILIAR WITH CYCLIC EXECUTIVES, DISTINGUISH BETWEEN
  - A TASK BEING "INITIATED" FOR ONE DUTY CYCLE AND THEN TERMINATING UNTIL ITS NEXT TURN, AND
  - THE Ada VIEW OF A PERIODIC TASK BEING ACTIVATED AT THE START OF A PROGRAM, EXECUTING A LOOP THAT IS REPEATED FOR EACH DUTY CYCLE
- IF CLASS DOES NOT CONTAIN EXPERIENCED REAL TIME PROGRAMMERS, THEN DO NOT SPEND MUCH TIME ON THIS POINT
- IF CLASS DOES CONTAIN EXPERIENCED REAL TIME PROGRAMMERS
  - MAKE SURE THEY UNDERSTAND THIS DISTINCTION
  - COULD RESULT IN A GREAT DEAL OF CONFUSION, OTHERWISE
- FOR BOTH DECLARED AND ALLOCATED TASKS, SHOW EXAMPLES OF ACTIVATION AND TERMINATION AND THEN PRESENT GENERAL RULES

INSTRUCTOR NOTES

- GIVE THE INSTRUCTORS IN TRAINING AN OVERVIEW OF WHAT IS COVERED IN PART III AND WHAT THE OBJECTIVES ARE

PART III - TASK COOPERATION

FOCUS OF UNIT:

- INTRODUCES SELECTIVE WAIT, TIMED ENTRY CALLS, AND CONDITIONAL ENTRY CALLS
    - SITUATIONS FOR WHICH EACH FORM IS APPROPRIATE
- DEADLOCK - HOW IT CAN OCCUR, AND WAYS TO AVOID IT
- EMPHASIS SHOULD BE ON
    - RENDEZVOUS AS THE BASIC MEANS OF Ada TASK COMMUNICATION
    - USER CONTROL OVER WHEN TASKS COMMUNICATE
    - POTENTIAL FOR DEADLOCK IF ATTENTION IS NOT PAID TO TASK COMMUNICATION PROTOCOL

SECTIONS:

- SECTION 7 - SIMPLE RENDEZVOUS (:45)
- SECTION 8 - SELECTIVE WAITS (2:00)
- SECTION 9 - SELECT STATEMENTS FOR MAKING ENTRY CALLS (:30)
- SECTION 10 - AVOIDING DEADLOCK (1:00)

INSTRUCTOR NOTES

- L401 STUDENTS NEED TO UNDERSTAND HOW A RENDEZVOUS IS ESTABLISHED

- INSTRUCTORS IN TRAINING SHOULD EMPHASIZE THE SIMILARITY BETWEEN ENTRIES AND SUBPROGRAMS AS OPERATIONS ON A TYPE

- L401 INSTRUCTORS SHOULD MAKE SURE THE CLASS UNDERSTANDS THE QUEUE MECHANISM
  - NEED THIS ALSO FOR 'Entry ATTRIBUTE
  - TIMED ENTRY CALLS

SECTION 7 - SIMPLE RENDEZVOUS

SUMMARY OF MAIN POINTS COVERED:

- INTRODUCE ENTRY CALLS, ACCEPT STATEMENTS, RENDEZVOUS

MAIN MESSAGES:

- THE RENDEZVOUS MECHANISM COMPLETELY DESCRIBES HOW Ada TASK COMMUNICATION WORKS
  - TASKS MUST AGREE TO COMMUNICATE
  - TASKS COMMUNICATE THROUGH PARAMETERS
- EACH TASK ENTRY HAS A QUEUE ASSOCIATED WITH IT TO HOLD CALLING TASKS

SUBTOPICS:

- REVIEW - ENTRY DECLARATIONS AND CALLS
- ACCEPT STATEMENT
- RENDEZVOUS

SPECIAL CONSIDERATIONS:

- EMPHASIZE THAT BOTH TASKS MUST AGREE TO COMMUNICATE
  - CALLER NAMES TASK IT WANTS TO RENDEZVOUS WITH
  - CALLED TASK DOES NOT NAME A TASK TO RENDEZVOUS WITH
  - THIS USER/SERVER RELATIONSHIP IS EXPLORED IN SECTION II
- EMPHASIZE THAT TASKS COMMUNICATE THROUGH PARAMETERS
  - STUDENTS MUST REALIZE TASKS WAIT UNTIL RENDEZVOUS CAN OCCUR AND THAT CALLING TASK WAITS WHILE CALLED TASK EXECUTES ACCEPT STATEMENT
  - STRESS THIS IN EACH EXAMPLE IN THIS SECTION
- MAKE SURE STUDENTS SEE SIMILARITY BETWEEN
  - ENTRY DECLARATIONS AND PROCEDURE DECLARATIONS
  - ENTRY CALLS AND PROCEDURE CALLS

  THIS HELPS THE STUDENTS THINK OF A TASK'S ENTRIES AS OPERATIONS ON THE TASK

INSTRUCTOR NOTES

SECTION 7 - EXERCISE

- THIS EXERCISE REVISITS SOUTHERN PATHETIC RAILROAD ONE LAST TIME
- THE PROBLEM IS MODIFIED TO REQUIRE THE STUDENTS TO WRITE A TRACK MANAGER TASK
  - THE TYPE DESCRIBED IN THE PREVIOUS SLIDE PROVIDES THE SOLUTION
  - THIS WILL GIVE YOU SOME IDEA AS TO HOW WELL THE CLASS IS FOLLOWING THE MATERIAL

INSTRUCTOR NOTES

- THE STUDENTS SHOULD UNDERSTAND THAT Ada PROVIDES THEM WITH WAYS TO CONTROL A CALLED TASK'S RENDEZVOUS
    - SOME L401 STUDENTS MIGHT COME TO THIS CLASS THINKING THAT Ada CANNOT POSSIBLY GIVE THEM ENOUGH CONTROL OVER A TASKS COMMUNICATION WITH OTHER TASKS
    - THIS SECTION, TOGETHER WITH THE NEXT, SHOULD MAKE THE STUDENTS AWARE THAT Ada IS A VIABLE LANGUAGE FOR CONTROLLING TASK COMMUNICATION FROM THE "CALLED" TASK'S POINT OF VIEW

- INSTRUCTORS IN TRAINING SHOULD UNDERSTAND THAT THIS SECTION AND THE NEXT ARE TWO OF THE MOST IMPORTANT SECTIONS IN THIS MODULE

SECTION 8 - SELECTIVE WAITS

SUMMARY OF MAIN POINTS COVERED:

- INTRODUCE FEATURES OF SELECTIVE WAITS AND SITUATIONS USING EACH
- GIVE STUDENTS BETTER UNDERSTANDING OF TASK TERMINATION

MAIN MESSAGES:

- TASKS CAN BE WRITTEN NONDETERMINISTICALLY
- TASKS CAN EXERCISE VARYING DEGREES OF CONTROL OVER THE ENTRY CALLS THEY ACCEPT AND WHEN THEY ACCEPT THEM
- TASKS CAN LIMIT OR AVOID WAITS FOR A RENDEZVOUS

SUBTOPICS:

- SELECT WAIT STATEMENT
- GUARDS
- TERMINATE ALTERNATIVE
- DELAY ALTERNATIVES
- ELSE PART

SPECIAL CONSIDERATIONS:

- IN DISCUSSING DELAY ALTERNATIVES, EMPHASIZE THAT THE ALTERNATE ACTION WILL TAKE PLACE SOMETIME AFTER THE SPECIFIED TIME, NOT "AT THAT MOMENT"
    - SECTION 14 COVERS THE PROBLEMS OF CUMULATIVE DRIFT AND JITTER; DO NOT MENTION HERE
- REMEMBER THAT EXPERIENCED REAL TIME PROGRAMMERS WILL BE CONCERNED ABOUT THE DEGREE OF CONTROL THEY HAVE OVER TASK COMMUNICATION
    - MAKE SURE STUDENTS UNDERSTAND THAT THIS SECTION IS ADDRESSING THE CALLED TASK'S VIEWPOINT
    - THE NEXT SECTION ADDRESSES THE CALLING TASK'S VIEWPOINT
- EACH FEATURE IS PRESENTED AS
    - HERE'S A REAL PROBLEM WE MIGHT NEED TO SOLVE
    - INTRODUCTION OF THE FEATURE NEEDED TO SOLVE THE PROBLEM
    - SOLUTION USING THE FEATURE
    - GENERAL RULES FOR THE FEATURES

INSTRUCTOR NOTES

SECTION 8 - EXERCISES

- FIVE EXERCISES FOR THIS SECTION
  - SIMPLY TEST UNDERSTANDING OF SELECTIVE WAIT STATEMENT
  - 10-30 MINUTES EACH
  - EACH EXERCISE MODIFIES THE PREVIOUS ONE
  - MAKE SURE YOU TELL CLASS TO LEAVE ROOM FOR CHANGES

- STARTS WITH SELECTIVE WAIT WITH SEVERAL ACCEPT ALTERNATIVES

- MODIFICATIONS INCLUDE:
  - ADD TERMINATE ALTERNATIVE
  - ADD DELAY ALTERNATIVES
  - NESTED SELECTIVE WAIT STATEMENTS

INSTRUCTOR NOTES

- IN L401 SECTION 8 STUDENTS WERE TAUGHT WAYS Ada ALLOWS THEM TO CONTROL A CALLED TASK'S RENDEZVOUS. THIS SECTION LOOKS AT THE OTHER DIRECTION.

- INSTRUCTORS IN TRAINING SHOULD REALIZE THAT UPON COMPLETION OF THIS SECTION L401 STUDENTS SHOULD RECOGNIZE Ada AS A VIABLE LANGUAGE FOR REAL TIME PROGRAMMING.

- WARN THE INSTRUCTORS IN TRAINING <u>NOT</u> TO CONCLUDE THAT ANY POSSIBLE TASK COMMUNICATION CAN BE WRITTEN IN Ada, OR THEY MIGHT BE ASKED HOW SOME "STRANGE" PROTOCOL CAN BE IMPLEMENTED.

SECTION 9 - SELECT STATEMENTS FOR MAKING ENTRY CALLS

SUMMARY OF MAIN POINTS COVERED:

- INTRODUCE TIMED ENTRY CALLS AND CONDITIONAL ENTRY CALLS
- ADDRESS CALLING TASK'S VIEWPOINT OF RENDEZVOUS

MAIN MESSAGES:

- A PROGRAM CAN SPECIFY ACTIONS TO BE TAKEN INSTEAD OF AN ENTRY CALL
  - IF THE ENTRY CALL CANNOT BE ACCEPTED IMMEDIATELY
  - IF THE ENTRY CALL CANNOT BE ACCEPTED WITHIN A SPECIFIED AMOUNT OF TIME
- FOR TIMED ENTRY CALLS, THE ALTERNATIVE ACTION OCCURS SOMETIME AFTER DELAY EXPIRES
  - DEPENDS ON RUNTIME SYSTEM

SUBTOPICS:

- TIMED ENTRY CALLS
- CONDITIONAL ENTRY CALLS
- COUNT ATTRIBUTE

SPECIAL CONSIDERATIONS:

- EACH FEATURE IS PRESENTED AS IN THE PREVIOUS SECTION
- FOR BOTH TIMED AND CONDITIONAL ENTRY CALLS
  - MAKE SURE CLASS REALIZES THAT EACH OF THESE STATEMENTS IS FOR A SINGLE ENTRY CALL
  - NESTED TIMED OR CONDITIONAL CALLS CAN BE USED TO SAY EXECUTE ONE OF THE FOLLOWING ENTRY CALLS, HOWEVER THE ORDER IS FIXED UNLIKE THE SELECTIVE WAIT STATEMENT
  - FOR SYNTAX, MAKE SURE THE CLASS REALIZES <u>or</u> IN TIMED ENTRY CALLS BUT else IN CONDITIONAL ENTRY CALLS
- EMPHASIZE THAT THE 'Count ATTRIBUTE SHOULD BE USED WITH GREAT CARE
  - IN PRESENCE OF TIMED ENTRY CALLS, THE CALLING TASK MUST
    - USE 'Count ONLY TO MEAN THAT ONE OR MORE TASKS ARE WAITING
    - USE A SELECTIVE WAIT WITH ELSE PART TO HANDLE CANCELLED CALLS AND ATTEMPTS TO USE 'Count AS ACTUAL COUNT WILL FAIL
  - DO NOT USE UNLESS ABSOLUTELY REQUIRED

INSTRUCTOR NOTES

- L401 STUDENTS SHOULD UNDERSTAND THAT THE FOUR NECESSARY CONDITIONS EXIST

SECTION 10 - AVOIDING DEADLOCK

SUMMARY OF MAIN POINTS COVERED:

- TWO POTENTIAL SOURCES FOR DEADLOCK
  - RESULTING FROM UNSAFE TASK COMMUNICATION PROTOCOL
  - IN RESOURCE SHARING
- FOUR NECESSARY CONDITIONS FOR DEADLOCK IN RESOURCE SHARING

MAIN MESSAGES:

- SAFE TASK COMMUNICATION PROTOCOL WILL NOT ALLOW DEADLOCK TO OCCUR
- UNSAFE TASK COMMUNICATION PROTOCOL NEED NOT RESULT IN DEADLOCK
- PIECEMEAL ALLOCATION CAN CAUSE DEADLOCK IN RESOURCE SHARING

SUBTOPICS:

- GUIDELINES FOR AVOIDING DEADLOCK
- RADAR SYSTEM EXAMPLE
- DEADLOCK DUE TO RESOURCE SHARING
  - NECESSARY CONDITIONS

SPECIAL CONSIDERATIONS:

- L401 INSTRUCTORS SHOULD EMPHASIZE THE POSSIBILITY OF DEADLOCK OCCURRING
ALWAYS EXISTS WHEN TASKS ARE COMMUNICATING, UNLESS CARE IS TAKEN TO AVOID IT
  - POSSIBILITY EXISTS EVEN WITH Ada
  - DEALING WITH DEADLOCK IS A DESIGN ISSUE
- RADAR EXAMPLE
  - DESCRIBES SEVERAL TASKS INVOLVED IN A RADAR SYSTEM
    - INITIAL ATTEMPT RESULTS IN DEADLOCK
    - REASONS FOR DEADLOCK ARE DISCUSSED
    - CORRECT SOLUTION IS GIVEN
  - IF TIME PERMITS, ASK THE CLASS TO FIND WHERE THE PROBLEM EXISTS
  - SHOWS WHAT CAN HAPPEN IF CARE IS NOT TAKEN IN ESTABLISHING A SAFE TASK COMMUNICATION PROTOCOL
  - GOOD ILLUSTRATION OF HOW DEADLOCK CAN "CREEP" INTO A SYSTEM

INSTRUCTOR NOTES

- GIVE THE INSTRUCTORS IN TRAINING AN OVERVIEW OF WHAT IS COVERED IN PART IV AND WHAT ITS OBJECTIVES ARE

SECTION IV - FUNDAMENTAL TASK DESIGNS

FOCUS OF UNIT:

- THIS PART OF THE COURSE DESCRIBES A VARIETY OF WAYS TO DESIGN TASKS AND DESIGN WITH TASKS
  - RELATIONSHIP BETWEEN TASKS
  - TASKS AS BUILDING BLOCKS FOR TASK COMMUNICATION SCHEMES
  - CYCLIC PROCESSING
  - SEQUENTIAL PROGRAM SOLUTIONS

SECTIONS:

- SECTION 11 - SERVER AND USER TASKS (:30)
- SECTION 12 - MONITORS (1:00)
- SECTION 13 - MESSAGE BUFFERS (:45)
- SECTION 14 - CYCLIC PROCESSING (1:00)
- SECTION 15 - STREAM-ORIENTED TASK DESIGN (1:30)

INSTRUCTOR NOTES

- MAKE SURE THE INSTRUCTORS IN TRAINING REALIZE THE DESIGN HINT AND THE EXAMPLE ILLUSTRATE WHY IT IS IMPORTANT TO UNDERSTAND USER AND SERVER TASKS

SECTION 11 - SERVER AND USER TASKS

SUMMARY OF MAIN POINTS COVERED:

- DISCUSS TASKS ACTING AS SERVERS VS. TASKS ACTING AS USERS

MAIN MESSAGES:

- ONE TASK CALLS ANOTHER IN ORDER TO OBTAIN SOME SERVICE FROM THAT TASK
- A TASK ACCEPTS AN ENTRY CALL IN ORDER TO PROVIDE SOME SERVICE TO THE CALLING TASK
- RENDEZVOUS ARE ASYMMETRIC
  - CALLING TASK NAMES THE TASK WHOSE ENTRY IT IS CALLING
  - CALLED TASK HAS NO WAY TO KNOW WHICH TASK'S ENTRY CALL IT IS ACCEPTING

SUBTOPICS:

- ENTRIES VS. PROCEDURES
- REVERSING DIRECTION OF RENDEZVOUS

SPECIAL CONSIDERATIONS:

- EMPHASIZE DESIGN HINT
  - DESIGN OF A MULTITASK PROGRAM CAN SOMETIMES BE SIMPLIFIED BY REVERSING THE DIRECTION OF RENDEZVOUS BETWEEN TWO TASKS
  - DIRECTION OF RENDEZVOUS SHOULD HIGHLIGHT USER/SERVER TASK ROLES
  - EXAMPLE OF MULTIZONE HEATING SYSTEM USED AS ILLUSTRATION OF THE EFFECT OF REVERSING THE DIRECTION OF THE RENDEZVOUS
- EMPHASIZE ANALOGY TO SUBPROGRAMS

INSTRUCTOR NOTES

- MAKE SURE INSTRUCTORS IN TRAINING REALIZE THEY SHOULD EMPHASIZE THE USE OF THE MONITOR FOR SAFELY SHARING DATA

SECTION 12 - MONITORS

SUMMARY OF MAIN POINTS COVERED:

- MONITORS IMPLEMENTED AS TASKS
- MONITORS FOR SHARING DATA AMONG TASK
- MONITORS AND PACKAGES

MAIN MESSAGES:

- A MONITOR ALLOWS OTHER TASKS TO SHARE DATA WITHOUT SIMULTANEOUS UPDATE PROBLEMS
- A PACKAGE WITH GLOBAL VARIABLES CAN NOT BE SHARED AMONG TASKS WITHOUT SPECIAL CONSIDERATION

SUBTOPICS:

- DEFINITION OF MONITOR
- TASK TYPES FOR MONITORS
- MONITORS AND PACKAGES

SPECIAL CONSIDERATIONS:

- EMPHASIZE
  - MONITORS AND PACKAGES HAVE CERTAIN PROPERTIES IN COMMON
    - EACH PROVIDES A LIMITED SET OF OPERATIONS THROUGH WHICH DATA HIDDEN INSIDE IT CAN BE MANIPULATED
    - PACKAGE GOOD FOR SINGLE TASK MANIPULATING THE DATA
    - MONITOR REQUIRED FOR SEVERAL TASKS MANIPULATING THE DATA
  - MONITOR CAN AND SHOULD BE THOUGHT OF AS A DATA OBJECT
  - GUARDS SHOULD BE USED TO ENFORCE THE PRECONDITIONS OF A DATA ABSTRACTION DEFINED AS A MONITOR
- AN EXISTING PACKAGE DESIGNED TO BE USED BY A SINGLE TASK MUST NOW BE USED BY SEVERAL TASKS, EMBED IT IN A MONITOR
- IF A DATA ABSTRACTION IS DEFINED AS A MONITOR, IT CAN BE EMBEDDED IN A PACKAGE TO PROVIDE THE SAME KIND OF EXTERNAL INTERFACE AS FOR NONSHARED DATA ABSTRACTIONS

INSTRUCTOR NOTES

SECTION 12 - EXERCISE

- THIS EXERCISE REQUIRES THE STUDENTS TO WRITE A PACKAGE PROVIDING OPERATIONS FOR SERVICING AN ELEVATOR
  - A CONTROL PROGRAM FOR THE ELEVATOR IS DESCRIBED
  - ALSO DESCRIBED IS A REQUEST TABLE USED BY THE CONTROL PROGRAM
  - SINCE THE CONTROL PROGRAM CAN BE CALLED UPON BY SEVERAL TASKS, A TASK IS REQUIRED TO BE WRITTEN

- THE EXERCISE CONSISTS OF TWO PARTS
  - PART 1: WRITE THE PACKAGE SPECIFICATION
    - STUDENTS MAY PLACE TASK DECLARATION IN THE PACKAGE SPECIFICATION OR MAY PROVIDE SUBPROGRAMS IN THE PACKAGE SPECIFICATION
    - FIRST APPROACH IS BETTER FOR THIS EXERCISE SINCE IT ELIMINATES A GREAT DEAL OF TEDIOUS WRITING IN PART 2
  - PART 2: WRITE THE PACKAGE BODY

INSTRUCTOR NOTES

- INSTRUCTORS IN TRAINING SHOULD REALIZE THAT THIS SECTION ILLUSTRATES HOW Ada RENDEZVOUS CAN BE USED TO PROVIDE MORE ELABORATE TASK COMMUNICATION PROTOCOLS
  - THIS WILL BE IMPORTANT TO MANY STUDENTS
- THE EXAMPLES ARE ALSO GOOD ILLUSTRATIONS OF COMBINING Ada TASK UNITS AND GENERIC UNITS

SECTION 13 - MESSAGE BUFFERS

SUMMARY OF MAIN POINTS COVERED:

- USE AND IMPLEMENTATION OF MESSAGE BUFFERS

MAIN MESSAGES:

- MESSAGE BUFFERS ALLOW TASKS TO COMMUNICATE WITHOUT WAITING FOR EACH OTHER
- RENDEZVOUS CAN BE USED AS A BUILDING BLOCK FOR MORE ELABORATE MECHANISMS FOR TASK COMMUNICATION
- MECHANISMS CAN BE DESIGNED ONCE AS GENERIC UNITS AND INSTANTIATED WHENEVER THE NEED ARISES

SUBTOPICS:

- SINGLE ELEMENT MESSAGE BUFFERS
- N-ELEMENT MESSAGE BUFFERS
- VERY FAST MESSAGE BUFFERS

SPECIAL CONSIDERATIONS:

- THE NEED FOR MESSAGE BUFFERS IS INTRODUCED BY USE OF A MULTIPLEXER DELIVERING A PACKET OF SENSOR READINGS TO A PROCESSING TASK
  - OCCASIONALLY THE PROCESSING TASK TAKES A LITTLE LONGER TO PROCESS THE PACKET, SO PACKET IS LOST
  - SINGLE ELEMENT MESSAGE BUFFER IS INTRODUCED AS THE SOLUTION
  - POSSIBILITY OF NEED TO BUFFER MORE THAN ONE PACKET IS RAISED
  - N-ELEMENT MESSAGE BUFFER IS INTRODUCED AS SOLUTION
- SOMETIMES CRITICAL OPERATIONS SUCH AS THE Send AND Receive OPERATIONS OF THE MESSAGE BUFFER MIGHT NEED TO BE IMPLEMENTED WITH CODE PROCEDURES, POSSIBLY USING FEATURES PROVIDED BY HARDWARE OR RUNTIME SYSTEM
  - NOT NORMAL APPROACH
  - SOMETIMES NEEDED

INSTRUCTOR NOTES

- MAKE SURE THE INSTRUCTORS IN TRAINING UNDERSTAND THAT EXPERIENCE IN REAL TIME PROGRAMMING IS NOT A PREREQUISITE

- CYCLIC EXECUTIVES ARE USED EXTENSIVELY IN REAL TIME PROGRAMMING, SO EXPERIENCED REAL TIME PROGRAMMERS WILL FIND THIS SECTION ESPECIALLY INTERESTING

SECTION 14 - CYCLIC PROCESSING

SUMMARY OF MAIN POINTS COVERED:

- DISCUSS CYCLIC PROCESSING AS REQUIRED IN MANY REAL TIME APPLICATIONS
- PROBLEMS OF CUMULATIVE DRIFT AND JITTER

MAIN MESSAGES:

- TRADITIONAL CYCLIC EXECUTIVE CAN BE IMPLEMENTED IN Ada
- BETTER APPROACHES ARE AVAILABLE
    - EASIER TO SCHEDULE
    - EASIER TO MODIFY
- Ada IS A VIABLE LANGUAGE FOR REAL TIME PROGRAMMERS

SUBTOPICS:

- MAJOR AND MINOR CYCLES
- TRADITIONAL APPROACH
- SINGLE-THREAD APPROACH
- SCHEDULING OF ACTIVITIES
- CUMULATIVE DRIFT
- JITTER

SPECIAL CONSIDERATIONS:

- THE PROBLEMS OF CUMULATIVE DRIFT AND JITTER ARE IMPORTANT TO REAL TIME PROGRAMMERS
    - SECTION GIVES PROGRAMMING HINT ON HOW TO DEAL WITH THESE PROBLEMS
    - MAKE SURE THE CLASS UNDERSTANDS THE SOLUTION

INSTRUCTOR NOTES

- MAKE SURE THE INSTRUCTORS IN TRAINING REALIZE
  - THE ARTICLE SHOULD BE READ BEFORE ATTEMPTING TO TEACH THIS SECTION FOR THE FIRST TIME
  - THIS SECTION DISCUSSES SOME TYPICAL REAL TIME PROGRAMMING PROBLEMS. AN INSTRUCTOR SHOULD NOT PASS HIMSELF/HERSELF OFF AS AN EXPERT IN REAL TIME PROGRAMMING

SPECIAL CONSIDERATIONS: (Continued)

- REMEMBER THAT EXPERIENCE IN REAL TIME PROGRAMMING IS NOT A PREREQUISITE FOR THIS MODULE
  - FOR EXPERIENCED REAL TIME PROGRAMMERS, MOST OF THE MATERIAL IS NOT NEW
    - SHOWS THESE PROGRAMMERS THAT WE UNDERSTAND THE KINDS OF PROBLEMS THEY MUST DEAL WITH
    - SHOWS THEY CAN USE Ada TO IMPLEMENT THE KINDS OF CYCLIC EXECUTIVES THEY ARE USED TO DEALING WITH
    - AT THE SAME TIME WE OFFER AN ALTERNATE APPROACH: SINGLE THREAD APPROACH
      - A DISTINCT TASK FOR EACH ACTIVITY
  - FOR L401 STUDENTS NOT FAMILIAR WITH REAL TIME PROGRAMMING, THE MATERIAL IS ALL NEW
    - THIS SECTION PROVIDES A GOOD INTRODUCTION TO REAL TIME PROGRAMMING FOR SUCH STUDENTS
    - BE CAREFUL, HOWEVER, SINCE THIS MATERIAL MIGHT SEEM FORMIDABLE TO SUCH STUDENTS
- FOR EXPERIENCED REAL TIME PROGRAMMERS, THERE IS A POTENTIAL FOR CONFUSION WHEN USING TASK IN THIS SECTION. MAKE SURE PROGRAMMERS UNDERSTAND THAT WE USE
  - TASK TO MEAN Ada TASK
  - ACTIVITY TO MEAN CYCLIC "TASK"
- L401 INSTRUCTORS SHOULD READ THE ARTICLE:

  Evolving Toward Ada in Real Time Systems

  BEFORE ATTEMPTING TO TEACH THIS SECTION
- THIS SECTION CAN BE SKIPPED IF
  - THE CLASS IS FALLING BEHIND, OR
  - THE BACKGROUND OF THE CLASS WARRANTS IT

  BUT DO SO ONLY IF NECESSARY SINCE
  - EXPERIENCED REAL TIME PROGRAMMERS WILL PROBABLY BE EAGER FOR THIS SECTION TO BE TAUGHT
  - GOOD INTRODUCTION FOR OTHER PROGRAMMERS

INSTRUCTOR NOTES

- THE INSTRUCTORS IN TRAINING SHOULD REALIZE THAT USING TASKING TO SOLVE A SEQUENTIAL PROBLEM WILL BE NEW TO MOST STUDENTS
  - THIS SECTION SHOWS HOW "NATURAL" THIS APPROACH CAN BE
- PRINCIPLES OF PROGRAM DESIGN BY M.A. JACKSON IS A GOOD REFERENCE FOR THIS SECTION
- SPECIAL CONSIDERATIONS, BULLET 1:
  - TELEGRAPH PROBLEM DEALS WITH COMPARING INCOMING MESSAGES ON TWO DIFFERENT CHANNELS TO INDICATE WHETHER OR NOT THEY ARE THE SAME

SECTION 15 - STREAM-ORIENTED TASK DESIGN

SUMMARY OF MAIN POINTS COVERED:

- STREAM TRANSFORMATIONS
- IMPLEMENTING DIFFICULT SEQUENTIAL PROBLEMS AS A SEQUENCE OF "EASIER" STREAM TRANSFORMATIONS
- Ada TASKS FOR IMPLEMENTING STREAM TRANSFORMATIONS

MAIN MESSAGES:

- STREAM-ORIENTED TASK DESIGN IS A USEFUL DESIGN TOOL
- STREAM TRANSFORMATIONS ARE INDIVIDUALLY SIMPLE
    - COMBINE TRANSFORMATIONS TO SOLVE MORE DIFFICULT PROBLEMS
    - USED TO OVERCOME STRUCTURE CLASH

SUBTOPICS:

- JACKSON STRUCTURED PROGRAMMING AND SYSTEM DEVELOPMENT
- DATA FLOW PROGRAMMING
- STREAM OPERATIONS IN Ada

SPECIAL CONSIDERATIONS:

- SECTION STARTS BY DESCRIBING JACKSON'S TELEGRAM PROBLEM
    - THIS IS A VERY DIFFICULT SEQUENTIAL PROBLEM
    - STUDENTS ARE ASKED TO THINK HOW THEY MIGHT SOLVE THIS PROBLEM
        - UNLESS TIME DOES NOT ALLOW IT, CLASS SHOULD BE GIVEN ABOUT 15 MINUTES TO ATTEMPT TO SOLVE THIS PROBLEM
        - ALLOWS CLASS TO APPRECIATE HOW DIFFICULT THE PROBLEM IS
        - STREAM ORIENTED SOLUTION IS GIVEN AT THE END OF THE SECTION

INSTRUCTOR NOTES

SPECIAL CONSIDERATIONS: (Continued)

- LINE REFORMATTING PROBLEMS
    - INTRODUCES PROBLEM OF STRUCTURE CLASH
    - TRADITIONAL SOLUTION GIVEN FOLLOWED BY OUTLINE OF STREAM-ORIENTED SOLUTION
    - STREAM-ORIENTED SOLUTION SHOWN TO BE EASILY IMPLEMENTED USING Ada TASKS
    - ENHANCEMENT TO THE LINE REFORMATTING PROBLEM SHOWS
        - HOW DIFFICULT IT IS TO ENHANCE THE TRADITIONAL SOLUTION
        - HOW EASY IT IS TO ENHANCE THE STREAM-ORIENTED SOLUTION

      FOR YOUR SAKE AND FOR THE STUDENTS' SAKE, DO NOT DISCUSS THE TRADITIONAL SOLUTION IN CLASS
        - INSTRUCTOR'S NOTES INCLUDES A DISCUSSION OF THE TRADITIONAL SOLUTION IN CASE CLASS "FORCES" YOU TO DISCUSS IT
        - REMEMBER, THE DISCUSSION IS A <u>SAFETY NET</u> FOR YOU

- EMPHASIZE STREAM-ORIENTED TASK DESIGN AS A USEFUL DESIGN TOOL
    - IT MAY INTRODUCE A LARGE NUMBER OF TASKS
    - FOR SOME IMPLEMENTATIONS THIS WILL BE A PROBLEM, WHILE FOR OTHERS, IT WILL NOT
    - THERE ARE MECHANICAL MANIPULATIONS FOR MERGING A SERIES OF STREAM TRANSFORMATIONS INTO A SINGLE TASK
        - DESCRIBED IN SECTION 24
        - MIGHT BE PERFORMED BY CLEVER COMPILERS

- TASK OPTIMIZATIONS ARE DISCUSSED IN SECTION 26, SO DO NOT LET YOURSELF GET INVOLVED IN EFFICIENCY DISCUSSIONS NOW

INSTRUCTOR NOTES

SECTION 15 - EXERCISE

- THIS EXERCISE REQUIRES THE STUDENTS TO WRITE A SEQUENCE OF STREAM TRANSFORMATIONS ON NAVIGATION DATA
  - THE EXERCISE CONSISTS OF THREE PARTS
  - PROBABLY BEST TO LIMIT AMOUNT OF TIME SPENT ON EACH PART
- PART 1
  - STUDENTS REQUIRED TO DRAW A STREAM/TRANSFORMATION DIAGRAM OF A SOLUTION TO THIS PROBLEM
  - MOST STUDENTS SHOULD SOLVE THIS PART WITHOUT DIFFICULTY, BUT JUST IN CASE, HAND OUT THE SOLUTION TO PART 1 BEFORE MOVING ON TO PART 2
- PART 2
  - STUDENTS REQUIRED TO WRITE TASK DECLARATIONS FOR EACH OF THE TRANSFORMATIONS
  - AGAIN, MOST OF THE STUDENTS SHOULD SOLVE THIS PART WITHOUT DIFFICULTY. HAND OUT THE SOLUTION TO PART 2 BEFORE MOVING ON TO PART 3
- PART 3
  - STUDENTS REQUIRED TO WRITE TASK BODIES FOR AS MANY OF THE TRANSFORMATIONS AS POSSIBLE
  - THIS IS WHERE SOME STUDENTS MAY HAVE DIFFICULTY
  - WHEN DISCUSSING THE SOLUTIONS, EMPHASIZE HOW NATURAL THEY ARE

INSTRUCTOR NOTES

- GIVE THE INSTRUCTORS IN TRAINING AN OVERVIEW OF WHAT IS COVERED IN PART V AND WHAT ITS OBJECTIVES ARE

PART V - OTHER TASKING FEATURES

FOCUS OF UNIT:

- DEALS WITH THE REMAINING TASKING FEATURES PROVIDED BY Ada

SECTIONS:

- SECTION 16 - ABORTING TASKS (:30)
- SECTION 17 - EXCEPTIONS IN MULTITASK PROGRAMS (:30)
- SECTION 18 - INTERRUPT ENTRIES (:30)
- SECTION 19 - ENTRY FAMILIES (:50)
- SECTION 20 - TASK PRIORITIES (:20)

INSTRUCTOR NOTES

- MAIN MESSAGES, BULLET #4:
    - CLEAN-UP ACTIONS MIGHT BE TO
        - FREE ACQUIRED RESOURCES
        - ENSURE CONSISTENCY OF A DATABASE
        - ETC.

SECTION 16 - ABORTING TASKS

SUMMARY OF MAIN POINTS COVERED:

- DISCUSS ABORT STATEMENT AND GUIDELINES FOR ITS USE

MAIN MESSAGES:

- THE ABORT STATEMENT DOES NOT ABORT A TASK
- THE ABORT STATEMENT SHOULD BE THOUGHT OF AS AN EMERGENCY BRAKE
- SHOULD ONLY BE USED IN EXTREME SITUATIONS, NOT ROUTINE TASK MANAGEMENT
- WHEN ONE TASK WANTS ANOTHER TASK TO TERMINATE, IT IS USUALLY DESIRABLE FOR THE SECOND TASK TO PERFORM CLEAN-UP ACTIONS BEFORE TERMINATING; THE ABORT STATEMENT DOES NOT PROVIDE FOR THIS
- A TASK SHOULD HAVE A "PLEASE TERMINATE" ENTRY IF ANOTHER TASK CAN REQUEST IT TO TERMINATE

SUBTOPICS:

- PROBLEMS WITH abort
- "PLEASE TERMINATE" ALTERNATIVE

SPECIAL CONSIDERATIONS:

- CAUTION: THE ABORT STATEMENT DOES NOT ABORT A TASK
  - IT MAKES THE TASK ABNORMAL
  - BE CAREFUL NOT TO SAY THE TASK IS ABORTED
- BE SURE THAT STUDENTS UNDERSTAND THE "EMERGENCY BRAKE" VIEW OF THE ABORT STATEMENT

INSTRUCTOR NOTES

- MAKE SURE THE INSTRUCTORS IN TRAINING REALIZE THAT THIS SECTION EXTENDS THE DISCUSSION OF EXCEPTIONS, STARTED IN L202 AND CARRIED THROUGH L305, TO TASKS.
  - STUDENTS SHOULD KNOW HOW EXCEPTION HANDLING WORKS IN PROGRAM UNITS OTHER THAN TASK UNITS
  - REVIEW OF EXCEPTION HANDLING SHOULD NOT BE NECESSARY

SECTION 17 - EXCEPTIONS IN MULTITASKING PROGRAMS

SUMMARY OF MAIN POINTS COVERED:

- RULES FOR RAISING AND PROPAGATING EXCEPTIONS IN MULTITASK PROGRAMS
- PROVIDES DESIGN GUIDELINES FOR EXCEPTIONS IN MULTITASK PROGRAMS

MAIN MESSAGES:

- SPECIAL RULES EXIST FOR GOVERNING
    - THE RAISING OF EXCEPTIONS DURING TASK COMMUNICATION
    - THE PROPAGATION OF EXCEPTIONS FROM ONE TASK TO ANOTHER
- BLOCK STATEMENTS MUST BE USED TO CAUSE AN ENTRY CALL TO RAISE AN EXCEPTION IN THE CALLING TASK BUT TO LEAVE THE CALLED TASK INTACT

SUBTOPICS:

- EXCEPTIONS IN TASK BODIES
    - DECLARATIVE PARTS
    - STATEMENT SEQUENCES
        - RAISED BY SELECTIVE WAIT
        - ARISING IN ACCEPT STATEMENT

SPECIAL CONSIDERATIONS:

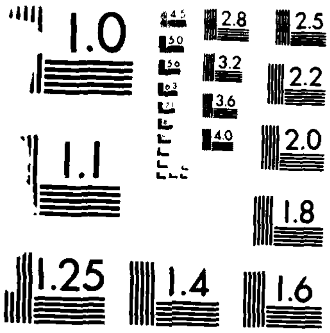
- THE STUDENTS SHOULD REALIZE THAT THE RULES FOR EXCEPTIONS IN MULTITASKING ARE NATURAL
    - RAISING AND PROPAGATING EXCEPTIONS BEHAVES EXACTLY THE WAY WE WOULD EXPECT THEM TO

INSTRUCTOR NOTES

- L401 INSTRUCTORS SHOULD MAKE SURE STUDENTS RECOGNIZE THAT THIS APPROACH TO INTERRUPT ENTRIES ALLOWS INTERRUPT HANDLERS TO BE WRITTEN AT A HIGHER LEVEL OF ABSTRACTION

AD-A165 346   ADA (TRADEMARK) TRAINING CURRICULUM INSTRUCTOR'S COURSE   3/3
5500 ADVANCED LANGUAGE MODULES(U) SOFTECH INC WALTHAM
MA 1986 DAAB07-85-C-K506

UNCLASSIFIED   F/G 5/9   NL



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

SECTION 18 - INTERRUPT ENTRIES

SUMMARY OF MAIN POINTS COVERED:

- DESCRIBE INTERRUPT ENTRIES AND DISCUSS THEIR USE

MAIN MESSAGES:

- A HARDWARE INTERRUPT CAN BE MADE TO LOOK TO AN Ada PROGRAM LIKE A CALL ON AN ENTRY
- THIS PROVIDES AN ABSTRACT VIEW OF HARDWARE INTERRUPTS AND ALLOWS DEVICE DRIVERS TO BE WRITTEN AT A HIGH LEVEL OF ABSTRACTION

SUBTOPICS:

- REVIEW ADDRESS CLAUSES
- PROPERTIES OF INTERRUPT ENTRIES
- EXAMPLE KEYBOARD INPUT HANDLER

SPECIAL CONSIDERATIONS:

- MAKE SURE STUDENTS RECOGNIZE THAT THIS APPROACH TO INTERRUPT ENTRIES ALLOWS INTERRUPT HANDLERS TO BE WRITTEN AT A HIGHER LEVEL OF ABSTRACTION
- ADDRESS CLAUSES ARE REVIEWED
    - NEEDED FOR ASSOCIATING ENTRY WITH HARDWARE INTERRUPT LOCATION
    - MAKE SURE YOU ARE COMFORTABLE WITH THE LOW-LEVEL FEATURES OF Ada WHEN TEACHING THIS SECTION
- CLASS SHOULD UNDERSTAND THAT USE OF INTERRUPT ENTRIES IS IMPLEMENTATION-DEPENDENT
    - THE TYPE System.Address
    - WHETHER INTERRUPTS ACT LIKE ORDINARY, TIMED OR CONDITIONAL ENTRY CALLS
    - WHETHER INTERRUPT ENTRIES HAVE PARAMETERS
- SERIAL INTERFACE DEVICE HANDLER EXAMPLE
    - ILLUSTRATES USE OF INTERRUPT ENTRIES
    - ALSO ILLUSTRATES USE OF EXCEPTIONS IN THE TASK

INSTRUCTOR NOTES

SECTION 18 - EXERCISE

- THIS EXERCISE REQUIRES THE STUDENTS TO WRITE A TASK WITH TWO ENTRIES
  - ONE IS CALLED BY TASK
  - THE OTHER IS AN INTERRUPT ENTRY
- THE TASK ITSELF IS SIMPLE MONITOR TASK, RECAST INTO A DIFFERENT SITUATION
  - TEST STUDENTS' UNDERSTANDING OF MONITORS
  - TESTS UNDERSTANDING OF INTERRUPT ENTRY

INSTRUCTOR NOTES

SECTION 19 - ENTRY FAMILIES

SUMMARY OF MAIN POINTS COVERED:

- INTRODUCE ENTRY FAMILIES
- USING ENTRY FAMILIES TO HANDLE REQUESTS HAVING DIFFERENT PRIORITIES

MAIN MESSAGES:

- A TASK MAY HAVE AN "ARRAY" OF ENTRIES, EACH OF WHICH HAS THE SAME FORMAL PART
- MEMBERS OF AN ENTRY FAMILY ARE INDEXED LIKE COMPONENTS OF A ONE-DIMENSIONAL ARRAY AND OPERATIONS ON ENTRIES CAN BE APPLIED TO INDIVIDUAL MEMBERS OF ENTRY FAMILIES

SUBTOPICS:

- BENEFITS OF ARRAYS IN PROGRAMMING
- ENTRY FAMILY DECLARATIONS
- ENTRY FAMILY CALLS

SPECIAL CONSIDERATIONS:

- MAKE SURE THE CLASS UNDERSTANDS THE DIFFERENCE BETWEEN AN ENTRY FAMILY AND AN ARRAY OF TASKS
    - THERE IS A GOOD CHANCE FOR CONFUSION HERE
    - POINT OUT THE DISTINCTION SEVERAL TIMES
- THE BUFFER ALLOCATION EXAMPLE SHOWS THE ALLOCATION OF BUFFERS BASED ON PRECEDENCE
    - FIRST APPROACH USES NEST SELECTIVE WAITS WITH ELSE PARTS
    - CLEANER SOLUTION SHOWN USING ENTRY FAMILY INDEXED BY AN ENUMERATION TYPE FOR PRECEDENCE
- WHEN TEACHING THIS SECTION, USE PRECEDENCE RATHER THAN PRIORITY SO AS NOT TO CAUSE CONFUSION WITH TASK PRIORITY (DISCUSSED IN THE NEXT SECTION)

INSTRUCTOR NOTES

- MAKE SURE THE INSTRUCTORS IN TRAINING UNDERSTAND THE FREQUENT CONFUSION ABOUT TASK PRIORITIES
    - L401 INSTRUCTORS SHOULD BE READY TO DEAL WITH THIS CONFUSION

SECTION 20 - TASK PRIORITIES

SUMMARY OF MAIN POINTS COVERED:

- EFFECT OF THE PRIORITY PRAGMA
- DEFINITION OF THE EFFECT OF TASK PRIORITY

MAIN MESSAGES:

- PRIORITIES ARE USED TO INDICATE RELATIVE DEGREES OF URGENCY
- PRIORITIES ARE USEFUL FOR RESOLVING CONTENTION AMONG TASKS FOR THE USE OF A CPU
- PRIORITIES ARE NOT USEFUL FOR TASK SYNCHRONIZATION

SUBTOPICS:

- PURPOSE OF PRIORITIES
- PRIORITY PRAGMA
- WHAT PRIORITIES DO/DON'T DO
- PRIORITIES AND
  - RENDEZVOUS
  - CYCLIC PROCESSING

SPECIAL CONSIDERATIONS:

- THERE IS OFTEN CONFUSION ON THE ROLE OF PRIORITIES
  - PEOPLE MISTAKENLY THINK PRIORITY DETERMINES WHICH TASK'S CALL ON AN ENTRY WILL BE ACCEPTED FIRST
    - STUDENTS NEED TO RECOGNIZE FIRST-IN/FIRST-OUT QUEUE FOR ENTRY DETERMINES THE ORDER OF ACCEPTANCE
  - PEOPLE MISTAKENLY THINK PRIORITY DETERMINES WHICH ALTERNATIVE OF A SELECTIVE WAIT IS CHOSEN (IMPLEMENTATION MIGHT USE PRIORITY, BUT IT MIGHT JUST AS WELL USE A STRATEGY EMPHASIZING FAIRNESS)
- TO UNDERSTAND THE ROLE OF PRIORITY IN THESE AREAS, THE MODULE PROVIDES
  - OFFICIAL DEFINITION OF THE EFFECT OF TASK PRIORITY
  - DISCUSSION OF WHAT THE DEFINITION MEANS
  - INTERPRETATION OF THE DEFINITION IN THE ABOVE AREAS

INSTRUCTOR NOTES

- GIVE THE INSTRUCTORS IN TRAINING AN OVERVIEW OF WHAT IS COVERED IN PART VI AND WHAT ITS OBJECTIVES ARE

- NOTE THAT WE HAVE POSTPONED TALKING ABOUT EFFICIENCY THROUGHOUT THE CURRICULUM UNTIL NOW

PART VI - IMPROVING PERFORMANCE

FOCUS OF UNIT:

- TUNING - WHEN TO TUNE, AND TECHNIQUES
- REDUCTION OF RENDEZVOUS FREQUENCY THROUGH SHARING OF GLOBAL VARIABLES
- MINIMIZING NUMBER OF TASKS AND TASK IDLE TIME
- TYPICAL COMPILER OPTIMIZATIONS

SECTIONS:

- SECTION 21 - WHEN TO TUNE (:45)
- SECTION 22 - SHARED VARIABLES (:30)
- SECTION 23 - MINIMIZING BLOCKING (:45)
- SECTION 24 - MERGING TASKS (:50)
- SECTION 25 - NON-CONCURRENT TUNING (2:00)
- SECTION 26 - WHAT'S BEST LEFT TO THE COMPILER (:45)

INSTRUCTOR NOTES

- INSTRUCTORS IN TRAINING SHOULD KEEP THIS SECTION LIGHT, BUT IT IS IMPORTANT FOR THEM TO
  - ENSURE THAT L401 STUDENTS UNDERSTAND WHEN TO TUNE AND HOW TO FIND OUT WHAT NEEDS TO BE TUNED
  - AVOID GIVING THE ERRONEOUS IMPRESSION THAT Ada IS INEFFICIENT

SECTION 21 - WHEN TO TUNE

SUMMARY OF MAIN POINTS COVERED:

- WHAT TUNING IS
- WHY IT IS NEEDED
- HOW TO DETERMINE WHEN TO TUNE

MAIN MESSAGES:

- TUNING IS OFTEN NECESSARY TO MEET REAL TIME CONSTRAINTS
- TYPICALLY, PROGRAMS SPEND MOST OF THEIR TIME EXECUTING A SMALL PART OF THE PROGRAM
  - TUNING MAKES SENSE ONLY IN THESE SMALL PARTS
  - PROGRAM MUST BE WORKING TO FIND THESE PARTS
  - NEED PROFILES TO FIND THEM
- PREMATURE TUNING MUST BE AVOIDED

SUBTOPICS:

- REAL TIME REQUIREMENTS
- HOW MUCH TO TUNE
- WHERE TO TUNE
- PREMATURE TUNING

SPECIAL CONSIDERATIONS:

- MANY OF THE MAJOR POINTS OF THIS SECTION ARE MADE THROUGH WAR STORIES
  - BOTH SUCCESS AND FAILURE
  - RELATE WAR STORIES YOU ARE FAMILIAR WITH
  - ENCOURAGE THE CLASS TO SHARE WAR STORIES THEY KNOW
- BE CAREFUL
  - SOME STUDENTS MIGHT THINK WE ARE SAYING Ada IS NOT EFFICIENT
  - SOME STUDENTS MIGHT FEEL THEY DO NOT NEED TO BE TAUGHT HOW TO TUNE THEIR CODE
- REMEMBER
  - STUDENTS NEED TO REALIZE WE UNDERSTAND THEIR CONCERN FOR EFFICIENCY
  - WE ARE SUGGESTING WAYS TO DEAL WITH IT
  - WE ARE ALSO MAKING THEM AWARE THAT A GOOD Ada COMPILER WILL GENERATE GOOD CODE

INSTRUCTOR NOTES

SECTION 22 - SHARED VARIABLES

SUMMARY OF MAIN POINTS COVERED:

- THE USE OF SHARED VARIABLES TO REDUCE TASK COMMUNICATION OVERHEAD
- THE SHARED PRAGMA

MAIN MESSAGES:

- TASKS CAN COMMUNICATE THROUGH SHARED VARIABLES AS WELL AS THROUGH ENTRY PARAMETERS
- SHARED VARIABLES CAN SOMETIMES BE MORE EFFICIENT BUT ALSO RISKY
- IN THE EXAMPLE, THE NEED FOR SHARED VARIABLES WAS DETERMINED BY A PROFILE

SUBTOPICS:

- REVIEW-TUNING
- MONITOR APPROACH
- RESULTS OF A PROFILE
- SHARED PRAGMA

SPECIAL CONSIDERATIONS:

- BE CAREFUL
    - SOME STUDENTS MAY THINK THAT Ada TASK COMMUNICATION IS INEFFICIENT OR THAT MONITORS ARE INEFFICIENT

INSTRUCTOR NOTES

SECTION 23 - MINIMIZING BLOCKING

SUMMARY OF MAIN POINTS COVERED:

- HOW TO REDUCE THE AMOUNT OF TIME TASKS ARE FORCED TO REMAIN IDLE

MAIN MESSAGES:

- IF TASKS SPEND MUCH OF THEIR TIME WAITING FOR OTHER TASKS OR DEVICES TO COMPLETE CERTAIN ACTIONS, PROCESSORS MAY BE UNDERUTILIZED
- WAITING TIME CAN BE REDUCED BY
  - KEEPING ACCEPT STATEMENTS SHORT
  - BUFFERING
  - USING LESS COARSE LOCKING STRATEGIES
  - SCHEDULING USE OF RESOURCES

SUBTOPICS:

- MAXIMIZING THROUGHPUT
- COMPUTATION WITHIN ACCEPT STATEMENTS
- BUFFERING
- SCHEDULING

SPECIAL CONSIDERATIONS:

- BUFFERING HAS ALREADY BEEN DISCUSSED SO DO NOT SPEND ANY TIME REVIEWING IT
- DO NOT GO INTO DEPTH INTO ANY OF THE CODE SECTIONS
  - JUST GIVE AN OVERVIEW OF THE CODE SECTION
  - POINT OUT MAJOR FEATURES, E.G., ENTRY FAMILY FOR SCHEDULING RESOURCE USE
- BE CAREFUL
  - SOME STUDENTS MAY THINK WE ARE SAYING ACCEPT STATEMENTS ARE INEFFICIENT

INSTRUCTOR NOTES

- THE INSTRUCTORS IN TRAINING SHOULD MAKE SURE THEY UNDERSTAND THE MERGING BEFORE TEACHING THIS SECTION. IN PARTICULAR, UNLESS AN L401 INSTRUCTOR TEACHES THIS SECTION FREQUENTLY, A FAIR AMOUNT OF PREPARATION TIME SHOULD BE DEVOTED TO THIS SECTION. THIS WILL ENABLE THE INSTRUCTOR TO HAVE THE DETAILS FRESH IN HIS/HER MIND, OTHERWISE, THE PRESENTATION WILL LOSE ITS IMPACT.

SECTION 24 - MERGING TASKS

SUMMARY OF MAIN POINTS COVERED:

- HOW TO MERGE SEPARATE TASKS INTO A SINGLE TASK

MAIN MESSAGES:

- STREAM-ORIENTED TASK DESIGN CAN BE USED EVEN IF MANY TASKS ARE INTRODUCED
- IF MANY TASKS EXIST, THE NUMBER CAN BE REDUCED THROUGH MERGING
- THE MERGED TASK CAN BE SUBSTANTIALLY LESS UNDERSTANDABLE THAN THE ORIGINAL TASKS
    - SOMETIMES HAPPENS WITH TUNING
    - STILL HAVE ORIGINAL TASK DESIGN AS DOCUMENTATION
- MERGING SHOULD ONLY BE PERFORMED IF NUMBER OF TASKS IS ESTABLISHED AS A PERFORMANCE PROBLEM

SUBTOPICS:

- COSTS OF HAVING MANY TASKS
- EASILY MERGEABLE TASKS
    - SIMPLE CASE
    - COMPLICATED CASE
- RESULT OF MERGING

SPECIAL CONSIDERATIONS:

- STUDENTS WILL BE ASKED TO MERGE TWO TASKS AS AN EXERCISE SO THEY NEED TO FULLY UNDERSTAND HOW MERGING (FOR THE SIMPLE CASE) WORKS
- AN L401 INSTRUCTOR MUST BE FULLY PREPARED WHEN TEACHING THIS SECTION
    - BE PREPARED TO POINT OUT THE EQUIVALENCE BETWEEN THE FLOW CHARTS AND THE Ada CODE

INSTRUCTOR NOTES

SECTION 24 - EXERCISE

- THIS EXERCISE REQUIRES THE STUDENTS TO MERGE TWO TASKS DEVELOPED EARLIER IN THE COURSE

- TWO PARTS TO THE EXERCISE

  - DRAW FLOWCHARTS FOR BOTH TASKS

  - DRAW FLOWCHART FOR THE MERGED TASK

  ENCOURAGE STUDENTS WHO FINISH EARLY TO WRITE THE RESULTING Ada CODE

- THE MERGING REQUIRES THE SIMPLE CASE

INSTRUCTOR NOTES

- INSTRUCTORS IN TRAINING SHOULD CONSIDER BENTLEY'S BOOK MUST READING.

SECTION 25 - NON-CONCURRENT TUNING

SUMMARY OF MAIN POINTS COVERED:

- TUNING TECHNIQUES THAT ARE APPLICABLE IN ANY TIME-CRITICAL (NOT NECESSARILY CONCURRENT) PROGRAM

MAIN MESSAGES:

- THERE ARE A VARIETY OF MEASURES FOR IMPROVING THE PERFORMANCE OF ONE TASK INDEPENDENTLY OF ITS RELATIONSHIP TO OTHER TASKS

SUBTOPICS:

- FUNDAMENTAL RULES
- ALGORITHM IMPROVEMENT
- SUBPROGRAM CALL IMPROVEMENT
  - INLINE PRAGMA
- REMOVING RECURSION
- LOOP UNFOLDING
- SUPPRESSING RUNTIME CHECKS
  - SUPPRESS PRAGMA
- SELECTIVE RECODING IN ASSEMBLY LANGUAGE

SPECIAL CONSIDERATIONS:

- THE CLASS MUST UNDERSTAND THE ASSUMPTIONS MADE THROUGHOUT THIS SECTION
  - PERFORMANCE PROBLEMS HAVE BEEN DETECTED IN THE PROGRAM PIECES WE LOOK AT
  - THE PROGRAM IS UNACCEPTABLE UNLESS "SUBSTANTIAL" IMPROVEMENT IS ACHIEVED IN THE PROGRAM PIECES
  - THESE PROGRAM PIECES HAVE BEEN FOUND THROUGH THE USE OF PROFILES

  THESE POINTS MUST BE EMPHASIZED DURING THE PRESENTATION SINCE
  - SOME OF THE TECHNIQUES USED GREATLY REDUCE READABILITY
  - WE DO NOT WANT STUDENTS TO THINK THE RESULTING CODE IS RECOMMENDED AS EXAMPLES OF GOOD PROGRAMMING STYLE
- BENTLEY'S BOOK, *Writing Efficient Programs* AND HIS COLUMN *Programming Pearls* (IN CACM) SHOULD BE RECOMMENDED TO THE CLASS
  - L401 INSTRUCTOR'S SHOULD CONSIDER THE BOOK AS A PREREQUISITE FOR THIS SECTION
  - L401 INSTRUCTOR'S SHOULD READ THE COLUMN ON A REGULAR BASIS

INSTRUCTOR NOTES

SECTION 26 - WHAT'S BEST LEFT TO THE COMPILER

SUMMARY OF MAIN POINTS COVERED:

- OPTIMIZATIONS THAT A PROGRAMMER CAN EXPECT FROM A GOOD OPTIMIZING COMPILER

MAIN MESSAGES:

- PREMATURE TUNING CAN MAKE A PROGRAM MORE DIFFICULT TO READ
    - MANY TIMES THE RESULTING CHANGES CAN BE MADE BY AN OPTIMIZING COMPILER
    - LOSS OF READABILITY DOES NOT OCCUR
- PROGRAMMERS SHOULD NOT MAKE UNWARRANTED ASSUMPTIONS ABOUT THE CODE GENERATED
- THE TASK OPTIMIZATIONS MIGHT NOT APPEAR IN THE EARLY Ada COMPILERS
    - STILL SHOULD NOT ASSUME THEY ARE NOT
    - EASIER TO TUNE AN UNDERSTANDABLE PROGRAM THAN TO TUNE WHILE WRITING A PROGRAM

SUBTOPICS:

- ROLE OF PRAGMAS
- TASK OPTIMIZATION
- COMMON SUBEXPRESSIONS
- CODE MOTION
- STRENGTH REDUCTION
- LOOP JAMMING
- DEAD CODE ELIMINATION

SPECIAL CONSIDERATIONS:

- POINT OUT THAT THE OPTIMIZATIONS PRESENTED IN THIS SECTION ARE REPRESENTATIVE OF THE OPTIMIZATIONS PERFORMED BY A GOOD OPTIMIZING COMPILER

INSTRUCTOR NOTES

PART VII - MODULE WRAP-UP

FOCUS OF UNIT:

- PROVIDES A COMPLETE REAL TIME PROGRAM WRITTEN IN Ada

SECTIONS:

- SECTION 27 - A COMPLETE EXAMPLE (1:30)

INSTRUCTOR NOTES

- THE INSTRUCTORS IN TRAINING SHOULD RECOGNIZE THAT THIS IS THE ONLY COMPLETE PROGRAM EXAMPLE USING Ada TASKING.

SECTION 27 - A COMPLETE EXAMPLE

SUMMARY OF MAIN POINTS COVERED:

- PRESENT A COMPLETE REAL TIME PROGRAM
- SHOW HOW TASKS FIT INTO A COMPLETE PROGRAM

MAIN MESSAGES:

- BY Ada STANDARDS, THIS PROGRAM IS MINISCULE (ONLY ABOUT 500 LINES), BUT IT ILLUSTRATES MANY OF THE MOST IMPORTANT IDEAS OF THE L401 MODULE

SUBTOPICS:

- EXAMPLE - MULTIZONE HEATING SYSTEM
  - REQUIREMENTS
  - TASKING STRUCTURE
  - PROGRAM UNIT STRUCTURE
  - PROGRAM CODE
- CONCLUSION

SPECIAL CONSIDERATIONS:

- THE PACE AND DEPTH WITH WHICH THIS SECTION IS TAUGHT CAN VARY, DEPENDING ON THE AMOUNT OF TIME LEFT. COMPLETE IN-DEPTH COVERAGE REQUIRES 1 1/2 HOURS.

INSTRUCTOR NOTES

L401 INSTRUCTORS BIBLIOGRAPHY

- The "Bug" Heard 'Round the World, John R. Garman ACM SIGSOFT SOFTWARE ENGINEERING NOTES, Vol. 6, No. 5 OCTOBER 1981, 3-10
  - DESCRIBES SOFTWARE BUG THAT POSTPONED FIRST FLIGHT OF THE SPACE SHUTTLE
  - SHOWS HOW DIFFICULT TASK SYNCHRONIZATION
  - BACKGROUND READING ONLY

- Evolving Toward Ada in Real Time Systems, Lee MacLaren ACM SIGPLAN NOTICES, Vol. 15, No. 11, NOVEMBER 1980, 146-155
  - DISCUSSES CYCLIC EXECUTIVES AND THEIR PROBLEMS
  - IMPLEMENTATION IN Ada
  - SHOULD READ THIS PAPER BEFORE TEACHING SECTION ON CYCLIC PROCESSING

- Principles of Program Design, M.A. Jackson, ACADEMIC PRESS, 1975
  - DISCUSSES STREAM TRANSFORMATION
  - DISCUSSES PROGRAM INVERSION - (TASK MERGING)
  - GOOD BACKGROUND READING

INSTRUCTOR NOTES

L401 INSTRUCTORS BIBLIOGRAPHY

- Programming Pearls, Jon L. Bentley, Communications of the ACM, BIMONTHLY FEATURE
  - WIDE RANGE OF PROGRAMMING TOPICS DISCUSSED
  - FREQUENTLY DISCUSSES EFFICIENT PROGRAMMING TECHNIQUES
  - SOURCE OF ADDITIONAL WAR STORIES
  - SHOULD READ REGULARLY
- Writing Efficient Programs, Jon L. Bentley, PRENTICE-HALL, 1982
  - EXCELLENT SOURCE FOR WRITING EFFICIENT PROGRAMS
  - DISCUSSES TUNING TECHNIQUES

INSTRUCTOR NOTES

- ALLOW 60 MINUTES FOR THIS SECTION

SECTION 4

L402

USING THE Ada LANGUAGE REFERENCE MANUAL

INSTRUCTOR NOTES

- GO OVER THIS SLIDE IN ENOUGH DETAIL TO GIVE THE INSTRUCTORS IN TRAINING AN OVERVIEW OF THIS MODULE.

- SECTION 4 SIMPLY LISTS THE CHAPTER TOPICS IN FORM SIMILAR TO THAT USED ON THIS SLIDE. THUS SECTION 4 WILL NOT BE MENTIONED IN THIS MODULE AGAIN.

- SECTIONS 5 THROUGH 18 DISCUSS THE 13 CHAPTERS OF THE LRM, BUT NOT IN THE ORDER LISTED IN THE LRM.
  - THE ORDER OF PRESENTATION IN L402 ALLOWS TOPICS TO BE DISCUSSED WITH A MINIMUM OF FORWARD REFERENCING
  - THESE SECTIONS ARE PRESENTED MAINLY AS EXERCISES
    - STUDENTS GET PRACTICE USING THE LRM
    - THE EXERCISE SOLUTIONS ARE DISCUSSED IN CLASS

- SECTION 19 WILL NOT BE MENTIONED IN THE MODULE AGAIN.

L402 COURSE OUTLINE

1. INTRODUCTION (1:00)
2. LANGUAGE REFERENCE MANUAL HISTORY (:30)
3. LRM STRUCTURE (1:00)
4. CHAPTER TOPICS (-)

5. LRM CHAPTER 1 - INTRODUCTION (:20)
6. LRM CHAPTER 2 - LEXICAL ELEMENTS (:10)
7. LRM CHAPTER 8 - VISIBILITY RULES (1:30)
8. LRM CHAPTER 3 - DECLARATIONS AND TYPES (1:30)
9. LRM CHAPTER 4 - NAMES AND EXPRESSIONS (:30)
10. LRM CHAPTER 7 - PACKAGES (:30)
11. LRM CHAPTER 6 - SUBPROGRAMS (:30)
12. LRM CHAPTER 9 - TASKS (1:00)
13. LRM CHAPTER 12 - GENERIC UNITS (:30)
14. LRM CHAPTER 11 - EXCEPTIONS (:30)
15. LRM CHAPTER 10 - PROGRAM STRUCTURE AND COMPILATION ISSUES (:30)
16. LRM CHAPTER 13 - REPRESENTATION CLAUSES AND IMPLEMENTATION-DEPENDENT FEATURES (:30)
17. LRM CHAPTER 5 - STATEMENTS (:25)
18. LRM CHAPTER - INPUT/OUTPUT (:55)

19. CONCLUSION (:10)

INSTRUCTOR NOTES

- SPECIAL CONSIDERATIONS, BULLET 1
  - INSTRUCTORS IN TRAINING SHOULD REALIZE THAT STUDENTS NOT SATISFYING THESE ASSUMPTIONS SHOULD NOT BE IN THE CLASS. ALL SUCH STUDENTS CAN DO IS SLOW DOWN BY ASKING INAPPROPRIATE QUESTIONS SUCH AS THOSE LISTED IN BULLET 2.

- MAIN MESSAGES, BULLETS 2-4
  - THE LRM MUST BE CONSIDERED THE SOURCE FOR CORRECT STATEMENT OF Ada SEMANTICS
    - IT HAS BEEN REVIEWED BY MANY PEOPLE, ORGANIZATIONS
    - THE IMPLEMENTERS' GUIDE SIMPLY HELPS READER TO "UNDERSTAND" THE LRM
  - TEXT BOOKS ARE ADEQUATE FOR LEARNING MOST OF THE Ada LANGUAGE, BUT
    - TEXT HAS NOT BEEN REVIEWED ENOUGH
    - TEXT WILL CONTAIN ERRORS
      - POSSIBLY DUE TO AUTHOR'S MISUNDERSTANDING
    - TEXT DOES NOT COVER ALL FEATURES IN DETAIL

SECTION 1 : INTRODUCTION

SUMMARY OF MAIN POINTS COVERED:

- WHAT THE STUDENTS SHOULD EXPECT TO GET OUT OF THIS COURSE
- LRM
- IMPLEMENTERS' GUIDE

MAIN MESSAGES:

- THIS MODULE DOES NOT TEACH Ada PROGRAMMING
- THE LRM IS THE ONLY COMPLETE REFERENCE FOR THE Ada PROGRAMMER
- THE IMPLEMENTERS' GUIDE, WHILE PRIMARILY FOR Ada IMPLEMENTERS, IS USEFUL FOR STUDENTS SINCE IT "EXPANDS" LRM DISCUSSIONS (TO PROVIDE MORE INFORMATION FOR IMPLEMENTING Ada)
- OTHER REFERENCES, EVEN A FAVORITE TEXTBOOK ON Ada, WILL BE INCOMPLETE AND PROBABLY INACCURATE

SUBTOPICS:

- COURSE TOPICS AND GOALS
- LRM STRUCTURE AND GOALS
- CHAPTER TOPICS
- NON-GOALS
- IMPLEMENTER'S GUIDE
- PREREQUISITES

INSTRUCTOR NOTES

SECTION 1 : INTRODUCTION - Continued

SPECIAL CONSIDERATIONS:

- MAKE SURE STUDENTS UNDERSTAND
  - THIS MODULE IS ABOUT USING THE LRM, NOT ABOUT USING Ada
  - EXPECTED THAT STUDENTS WILL BE ANSWERING QUESTIONS FOR OTHERS
    - NEED TO HAVE COMPLETE DETAILED UNDERSTANDING OF LANGUAGE ISSUES
    - ULTIMATELY MUST BE TO ANSWER: "WHAT'S WRONG WITH THIS PROGRAM?"
  - EXPECTED TO HAVE A STRONG BACKGROUND IN Ada

    PACKAGES EXCEPTIONS

    PRIVATE TYPES LIMITED TYPES

    TASKS GENERICS

  THIS MODULE DOES NOT TEACH Ada PROGRAMMING
- MAKE SURE CLASS UNDERSTANDS FOLLOWING QUESTIONS NOT APPROPRIATE IN L402
  - WHEN SHOULD A PARTICULAR FEATURE BE USED
  - WHY IS THIS FEATURE DEFINED THIS WAY
  - HOW DO I IMPLEMENT THIS IN Ada

INSTRUCTOR NOTES

SECTION 2 : LANGUAGE REFERENCE MANUAL HISTORY

SUMMARY OF MAIN POINTS COVERED:

- LRM HISTORY (NOT Ada LANGUAGE HISTORY)
- LRM REWRITE
- EXAMPLE OF REVIEW COMMENTS
- Ada BOARD

MAIN MESSAGES:

- THE WORDING IN THE LRM IS INTENDED TO BE PRECISE
    - LRM WAS REWRITTEN TO ACCOMPLISH THIS
    - REVIEW OF LRM PERFORMED BY PEOPLE FROM MANY PARTS OF THE WORLD
    - RESULTING LRM PROVIDES A MORE RIGOROUS LANGUAGE DEFINITION
- Ada BOARD EXISTS TO HANDLE
    - Ada QUESTIONS
    - INTERPRETATIONS OF LRM

SUBTOPICS:

- LRM HISTORY
- THE LRM REWRITTEN
- REVIEW COMMENTS EXAMPLE
- Ada BOARD

SPECIAL CONSIDERATIONS:

- EXAMPLE OF REVIEW COMMENTS FOR LRM REWRITE
    - ILLUSTRATES THE TECHNICAL QUESTIONS ASKED
    - SHOWS TWO VIEWPOINTS ON A TECHNICAL QUESTION
        - TWO COMMENTS
        - RESPONSES
    - ILLUSTRATES ROLE OF LANGUAGE DESIGN TEAM IN RESOLVING QUESTIONS

INSTRUCTOR NOTES

- THE EXERCISES ARE SHORT AND "SIMPLE".

SECTION 3 : LRM STRUCTURE

SUMMARY OF MAIN POINTS COVERED:

- NOTATION AND TERMS USED IN THE LRM
- STRUCTURE OF THE LRM

MAIN MESSAGES:

- NEED TO UNDERSTAND STRUCTURE OF THE LRM TO USE IT EFFECTIVELY
- SYNTAX IS NOT ENOUGH TO SETTLE LEGALITY
  - MUST UNDERSTAND RESTRICTIONS
  - MUST KNOW TECHNICAL TERMS (PRESENTED IN L402 AS NEEDED)

SUBTOPICS:

- SYNTAX
- NOTES AND EXAMPLES
- LANGUAGE TERMS
- REFERENCES
- ANNEXES AND APPENDICES

SPECIAL CONSIDERATIONS:

- STUDENTS MUST BE ABLE TO
  - READ BNF VARIANT USED IN LRM
    - INCLUDING TYPE FACE CONVENTIONS
  - SHOULD KNOW WHERE TO FIND COMPLETE SYNTAX (APPENDIX E)
  - UNDERSTAND THE PURPOSE OF NOTES AND EXAMPLES
  - UNDERSTAND PURPOSE OF REFERENCES AT THE END OF EACH LRM SECTION
- STUDENTS SHOULD UNDERSTAND WHAT INFORMATION IS AVAILABLE IN THE ANNEXES AND APPENDICES
  - ANNEX A - PREDEFINED LANGUAGE ATTRIBUTES
  - ANNEX B - PREDEFINED LANGUAGE PRAGMAS
  - ANNEX C - PREDEFINED LANGUAGE ENVIRONMENT
  - APPENDIX D - GLOSSARY
  - APPENDIX E - SYNTAX SUMMARY AND SYNTAX CROSS REFERENCE
  - APPENDIX F - IMPLEMENTATION-DEPENDENT CHARACTERISTICS

INSTRUCTOR NOTES

- THE EXERCISES ARE SHORT AND "SIMPLE"

SECTION 3 : LRM STRUCTURE - Continued

EXERCISES:

- ALLOW A TOTAL OF 55 MINUTES
- FOUR EXERCISES
  - FIRST TWO EXERCISES TEST ABILITY TO USE SYNTAX RULES
    - FIRST GIVES SYNTAX RULES NEEDED
    - SECOND REQUIRES USES OF APPENDIX E
  - LAST TWO EXERCISES FORCE THE STUDENTS TO USE THE ANNEXES

INSTRUCTOR NOTES

- MENTION, JUST THIS TIME, THAT EACH SECTION BEGINS WITH A LIST OF SECTION TOPICS (THE TABLE OF CONTENTS FOR THE CHAPTER)

- SPECIAL CONSIDERATIONS:

  - AN EXAMPLE ILLUSTRATING THE DIFFERENCE BETWEEN EXECUTION TIME ERRORS AND ERRONEOUS EXECUTION

```
  ...
  A : array (1 .. 10) of Integer;
  I : Integer := 12;
  ...
begin
  A (I) := 7;
     ...
```

NORMALLY THIS ASSIGNMENT WOULD RESULT IN AN EXECUTION TIME ERROR WITH THE EXCEPTION Constraint_Error RAISED. HOWEVER, IN THE PRESENCE OF

```
pragma Suppress (Index_Check, on => A)
```

THE EXECUTION IS ERRONEOUS SINCE THE Suppress PRAGMA IN THIS CASE WAS INFORMATION TO THE COMPILER THAT INDEX CHECKING IS NOT NEEDED

SECTION 5 : LRM CHAPTER 1 - INTRODUCTION

SUMMARY OF MAIN POINTS COVERED:

- DEFINITION OF A CONFORMING COMPILER
- DESIGN GOALS
- ERROR CATEGORIES
- IMPORTANT DEFINITIONS

MAIN MESSAGES:

- CHAPTER 1 CONTAINS GENERAL INFORMATION ABOUT THE LRM

SUBTOPICS:

- CONFORMING COMPILER
- DESIGN GOALS
- ERROR CATEGORIES
  - COMPILE TIME ERRORS
  - EXECUTION ERRORS
  - ERRONEOUS EXECUTION
  - INCORRECT ORDER DEPENDENCIES
- DEFINITIONS
  - LEGAL/ILLEGAL
  - ERRONEOUS
  - MUST
  - ALLOWED

SPECIAL CONSIDERATIONS:

- STUDENTS NEED TO UNDERSTAND THE FOUR ERROR CATEGORIES
  - COMPILE TIME ERRORS AND EXECUTION TIME ERRORS MUST BE DETECTED BY AN IMPLEMENTATION
  - ERRONEOUS EXECUTION AND INCORRECT ORDER DEPENDENCIES NEED NOT BE DETECTED BY AN IMPLEMENTATION

EXERCISES:

- ALLOW 5 MINUTES
- SHORT PROBLEM ON ORDER DEPENDENCY

INSTRUCTOR NOTES

- SPECIAL CONSIDERATIONS, BULLET 2
  - THE MATERIAL IN CHAPTER 2 IS EASY ENOUGH FOR THE STUDENTS TO UNDERSTAND ON THEIR OWN.
  - DO NOT SKIP UNLESS NEED TO! REMEMBER ONLY PICKING UP 10 MINUTES.

- EXERCISES
  - AGAIN, THESE EXERCISES ARE SHORT AND SIMPLE
    - EXAMPLES
      - IS -3 AN INTEGER LITERAL? (NO)
      - ARE Integer, integer, AND INTEGER ALL CONSIDERED TO BE THE SAME IDENTIFIER?

SECTION 6 : LRM CHAPTER 2 - LEXICAL ELEMENTS

SUMMARY OF MAIN POINTS COVERED:

- PRAGMAS

MAIN MESSAGES:

- CHAPTER 2 CONTAINS THE RULES FOR LEXICAL ELEMENTS AND PRAGMAS

SPECIAL CONSIDERATIONS:

- STUDENTS MIGHT NOT BE EXPECTING TO FIND RULES FOR PRAGMAS WITH THOSE FOR LEXICAL ELEMENTS
- THIS IS AN EASY CHAPTER AND MAY BE SKIPPED IF YOU ARE BEHIND SCHEDULE

EXERCISES:

- ALLOW 5 MINUTES
- TWO EXERCISES
  - TESTS STUDENT'S ABILITY TO USE LRM CHAPTER 2 TO ANSWER QUESTIONS ABOUT LEXICAL ELEMENTS
  - ANSWERS MUST BE SUPPORTED WITH RELEVANT LRM SECTION AND PARAGRAPH NUMBERS

INSTRUCTOR NOTES

- HOMOGRAPHS MAY BE NEW TO SOME OF THE INSTRUCTORS IN TRAINING. IF FORCED TO REVIEW HOMOGRAPHS FOR THEM, THE FOLLOWING DESCRIPTION FROM L402 CAN BE USED.
  - HOMOGRAPHS
    - DECLARATIONS WITH THE SAME IDENTIFIER AND WHERE AT MOST ONE IS OVERLOADABLE
    - DECLARATIONS WITH THE SAME IDENTIFIER, OPERATOR SYMBOL, OR CHARACTER LITERAL, AS WELL AS THE SAME PARAMETER AND RESULT PROFILE WHERE BOTH ARE OVERLOADABLE

SECTION 7 : LRM CHAPTER 8 - VISIBILITY RULES

SUMMARY OF MAIN POINTS COVERED:

- LRM ISSUES RELATED TO VISIBILITY RULES

MAIN MESSAGES:

- CHAPTER 8 CONTAINS THE RULES FOR VISIBILITY
  - UNDERSTANDING THIS CHAPTER IS FUNDAMENTAL TO UNDERSTANDING MOST OF THE REST OF THE LRM

SUBTOPICS:

- ENTITIES WITH AN EXTENDED SCOPE
- HOMOGRAPHS
- ENTITIES FOR WHICH OVERLOADING IS DEFINED
- IMPORTANT DEFINITIONS
  - LOCAL/GLOBAL
  - IMMEDIATE SCOPE
  - VISIBLE/HIDDEN

SPECIAL CONSIDERATIONS:

- HOMOGRAPHS
  - THIS MATERIAL WILL BE NEW TO MOST STUDENTS
  - CLASS MUST UNDERSTAND THAT A HOMOGRAPH IS USED IN DECIDING WHETHER ONE DECLARATION
    - HIDES ANOTHER
    - OVERLOADS ANOTHER

EXERCISES:

- ALLOW 30 MINUTES TOTAL
- TWO EXERCISES
  - STUDENTS MUST USE LRM CHAPTER 8 TO ANSWER VISIBILITY QUESTIONS
  - EXAMPLES
    - GIVEN A PROGRAM FRAGMENT, DEFINE THE DECLARATIVE REGION FOR A PARTICULAR OBJECT
    - DETERMINE THE HOMOGRAPHS IN AN EXAMPLE
  - ANSWERS MUST BE SUPPORTED WITH RELEVANT LRM SECTION AND PARAGRAPH NUMBERS

INSTRUCTOR NOTES

- SOME OF THE IMPORTANT DEFINITIONS INCLUDE ELABORATION, UNIVERSAL_REAL, AND UNIVERSAL_INTEGER

- EXERCISES
  - EXAMPLES
    - WHY ARE

      ```
      X, Y, Z : array (1 .. 20) of Boolean;
      ```

      CONSIDERED TO BE THREE DIFFERENT TYPES?

    - WHY IS THE FOLLOWING ILLEGAL?

      ```
      type A is array (Integer range <> , Boolean) of Character;
      ```

SECTION 8 : LRM CHAPTER 3 - DECLARATIONS AND TYPES

SUMMARY OF MAIN POINTS COVERED:

- ● DEFINITIONS AND CONCEPTS RELATED TO TYPE AND OBJECT DECLARATIONS

MAIN MESSAGES:

- ● CHAPTER 3 CONTAINS THE RULES FOR DECLARATIONS OF OBJECTS AND OF ALL TYPES EXCEPT FOR PRIVATE TYPES AND TASK TYPES
- ● WHEN THE LRM REFERS TO A TYPE IT MEANS A BASE TYPE

SUBTOPICS:

- ● TYPE DEFINITIONS
  - PARENT TYPE/SUBTYPE
  - DERIVED TYPE/SUBTYPE
  - BASE TYPE
- ● NAME AND DENOTE
- ● DESIGNATED ENTITIES
- ● IMPORTANT DEFINITIONS

SPECIAL CONSIDERATIONS:

- ● EMPHASIZE THAT NAME AND DENOTE ARE USED VERY DELIBERATELY IN THE LRM
  - NAME IS A SYNTACTIC CATEGORY
  - DENOTE MEANS TO NAME SOME ENTITY
- ● SUGGEST THAT STUDENTS LOOK TO THE IMPLEMENTERS' GUIDE FOR FURTHER INFORMATION

EXERCISES:

- ● ALLOW 50 MINUTES TOTAL
- ● TWO EXERCISES
  - STUDENTS MUST USE LRM CHAPTER 3 TO ANSWER QUESTIONS ABOUT OBJECT DECLARATIONS AND TYPE DECLARATIONS
  - ANSWERS MUST BE SUPPORTED WITH RELEVANT LRM SECTION AND PARAGRAPH NUMBERS

INSTRUCTOR NOTES

SECTION 9 : LRM CHAPTER 4 - NAMES AND EXPRESSIONS

SUMMARY OF MAIN POINTS COVERED:

- DISCUSS PREFIXES, DEFINITION OF STATIC

MAIN MESSAGES:

- CHAPTER 4 CONTAINS THE RULES FOR NAMES AND EXPRESSIONS
- MANY RULES EXIST TO MAKE THE LANGUAGE CONSISTENT

SUBTOPICS:

- PREFIX
    - APPROPRIATE FOR A TYPE
- DEFINITION OF STATIC
    - STATIC ENTITIES

SPECIAL CONSIDERATIONS:

- CLASS SHOULD UNDERSTAND
    - WHAT A PREFIX IS
        - IN PARTICULAR, THEY SHOULD UNDERSTAND WHAT THE LRM MEANS BY "A PREFIX THAT IS APPROPRIATE FOR A TYPE"
    - MANY RULES STATE THAT AN ENTITY MUST BE STATIC, SO THEY NEED TO UNDERSTAND
        - STATIC ENTITIES
        - STATIC SUBTYPES

EXERCISES:

- ALLOW 20 MINUTES TOTAL
- TWO EXERCISES
    - STUDENTS REQUIRED TO USE LRM CHAPTER 3, 4 AND 8 TO ANSWER QUESTIONS ABOUT NAMES AND EXPRESSIONS
    - EXAMPLES
        - DETERMINE LEGALITY OF AN EXPANDED NAME
        - DETERMINE LEGALITY OF AGGREGATES
    - ANSWERS MUST BE SUPPORTED WITH RELEVANT LRM SECTION AND PARAGRAPH NUMBERS

INSTRUCTOR NOTES

- EXERCISES

  - EXAMPLES

    - STUDENTS SHOWN A PRIVATE TYPE WITH FULL TYPE DECLARATION IN PACKAGE BODY
    - STUDENTS SHOWN A DEFERRED CONSTANT IN PACKAGE NOT CONTAINING A PRIVATE TYPE

    STUDENTS ARE ASKED TO EXPLAIN WHY THESE ARE ILLEGAL

SECTION 10 : LRM CHAPTER 7 - PACKAGES

SUMMARY OF MAIN POINTS COVERED:

- EXERCISES AND DEFINITIONS RELATED TO PACKAGES AND PRIVATE TYPES

MAIN MESSAGES:

- CHAPTER 7 CONTAINS THE RULES FOR PACKAGES AND PRIVATE TYPES

SUBTOPICS:

- IMPORTANT DEFINITIONS
    - VISIBLE PART
    - PRIVATE PART

EXERCISES:

- ALLOW 30 MINUTES TOTAL
- TWO EXERCISES
    - STUDENTS REQUIRED TO USE LRM CHAPTER 7 TO ANSWER QUESTIONS ABOUT
        - PRIVATE TYPES
        - LIMITED TYPES
        - DEFERRED CONSTANTS
    - ANSWERS MUST BE SUPPORTED WITH RELEVANT LRM SECTION AND PARAGRAPH NUMBERS

INSTRUCTOR NOTES

- IMPORTANT DEFINITIONS ARE
  - READ - WHEN AN OBJECT'S VALUE IS EVALUATED
  - UPDATED - WHEN AN ASSIGNMENT IS PERFORMED ON A VARIABLE

- EXERCISES
  - EXAMPLES
    - DETERMINE IF TWO GIVEN PROCEDURE SPECIFICATIONS CONFORM
    - STUDENTS GIVEN AN EXAMPLE OF A PROGRAM WHOSE RESULT DEPENDS ON THE PARAMETER PASSING TECHNIQUE, AND ASKED TO EXPLAIN WHERE THE DEPENDENCE IS AND WHETHER THE PROGRAM IS ERRONEOUS

SECTION 11 : LRM CHAPTER 6 - SUBPROGRAMS

SUMMARY OF MAIN POINTS COVERED:

- CONFORMANCE RULES AND DEFINITIONS RELATED TO SUBPROGRAMS

MAIN MESSAGES:

- CHAPTER 6 CONTAINS THE RULES FOR FUNCTIONS AND PROCEDURES

SUBTOPICS:

- CONFORMANCE RULES
  - DEFINITIONS
  - VARIATIONS
- IMPORTANT DEFINITIONS
  - READ
  - UPDATED

SPECIAL CONSIDERATIONS:

- QUICK SUMMARY OF CONFORMANCE RULES IS GIVEN
  - THROUGHOUT LRM, IT SAYS THINGS MUST CONFORM
  - THIS IS THE ONLY PLACE IN THE LRM THAT STATES WHAT IT MEANS
  - L402 INSTRUCTORS SHOULD READ LRM SECTION 6.3 (1) Conformance Rules BEFORE TEACHING THIS SECTION

EXERCISES:

- ALLOW 25 MINUTES TOTAL
- TWO EXERCISES
  - STUDENTS REQUIRED TO USE LRM CHAPTER 6 TO ANSWER QUESTIONS ABOUT PARAMETER PASSING, CONFORMANCE OF TYPES AND SUBPROGRAMS
- ANSWERS MUST BE SUPPORTED WITH RELEVANT LRM SECTION AND PARAGRAPH NUMBERS

INSTRUCTOR NOTES

- LAST BULLET
  - ITEM 1 - THESE TASKING QUESTIONS ARE SIMPLE
- THE IMPORTANT DEFINITIONS INCLUDE MASTER, COMPLETED, ABNORMAL

SECTION 12 : LRM CHAPTER 9 - TASKS

SUMMARY OF MAIN POINTS COVERED:

- CONCEPTS AND DEFINITIONS RELATED TO TASKING

MAIN MESSAGES:

- LRM CHAPTER 9 CONTAINS THE RULES FOR Ada TASKING
    - MANY THINGS ARE LEFT UNSPECIFIED BECAUSE NO EXPLICIT ORDER IS DESIRED
        - ALLOWS TASKS TO BE WRITTEN NONDETERMINISTICALLY
        - ALLOWS TAILORING OF RUNTIME SYSTEM

SUBTOPICS:

- DEPENDENCE
- TASK TERMINATION
- SHARED VARIABLES
- SYNCHRONIZATION POINTS
- IMPORTANT DEFINITIONS

SPECIAL CONSIDERATIONS:

- THE TOPICS DISCUSSED IN THIS SECTION ARE COVERED IN BOTH
    - L303 - REAL TIME CONCEPTS
    - L401 - REAL TIME SYSTEMS IN Ada

  IF STUDENTS HAVE TAKEN EITHER MODULE, THEN JUST REVIEW THIS MATERIAL; OTHERWISE SPEND THE FULL AMOUNT OF TIME ALLOTTED

EXERCISES:

- ALLOW 40 MINUTES TOTAL
- TWO EXERCISES
    - STUDENTS REQUIRED TO USE LRM CHAPTER 9 TO ANSWER TASKING QUESTIONS
    - EXAMPLES
        - GIVEN PROCEDURE WITH DECLARED AND ALLOCATED TASKS, STUDENTS ASKED WHEN TASKS START
    - ANSWERS MUST BE SUPPORTED WITH RELEVANT LRM SECTION AND PARAGRAPH NUMBERS

INSTRUCTOR NOTES

- THE IMPORTANT DEFINITIONS INCLUDE
  - TEMPLATE - CONSTRUCTION OR BUILDING TOOL
  - INSTANCES - OCCURRENCE OR COPY

- EXERCISES
  - EXAMPLES
    - STUDENTS PRESENTED WITH AN EXAMPLE OF AN ILLEGAL GENERIC FORMAL TYPE DECLARATION AND ASKED TO EXPLAIN WHY THEY ARE ILLEGAL
    - STUDENTS PRESENTED WITH AN EXAMPLE OF AN ILLEGAL INSTANTIATION AND ARE ASKED TO EXPLAIN WHY

SECTION 13 : LRM CHAPTER 12 - GENERIC UNITS

SUMMARY OF MAIN POINTS COVERED:

- EXERCISES AND DEFINITIONS RELATED TO GENERIC UNITS

MAIN MESSAGES:

- CHAPTER 7 CONTAINS THE RULES FOR GENERIC UNITS AND INSTANTIATIONS

SUBTOPICS:

- IMPORTANT DEFINITIONS
    - TEMPLATE
    - INSTANCES
    - MATCH

EXERCISES:

- ALLOW 25 MINUTES TOTAL
- TWO EXERCISES
    - STUDENTS REQUIRED TO USE LRM CHAPTER 12 TO ANSWER QUESTIONS ABOUT
        - DECLARATIONS OF GENERIC UNITS
        - GENERIC INSTANTIATION
    - ANSWERS MUST BE SUPPORTED WITH RELEVANT LRM SECTION AND PARAGRAPH NUMBERS

INSTRUCTOR NOTES

- EXERCISES
  - EXAMPLES
    - STUDENTS PRESENTED WITH AN EXAMPLE OF A FUNCTION CALL THAT RESULTS IN THE FUNCTION COMPLETING WITHOUT EXECUTING A RETURN. THE STUDENTS ARE ASKED TO EXPLAIN WHY Program_Error HAS BEEN RAISED
    - STUDENTS SHOWN AN EXAMPLE CONTAINING

      ```
      pragma Suppress (Range_Check);
      ```

      BUT THAT STILL HAS Constraint_Error RAISED

SECTION 14 : LRM CHAPTER 11 - EXCEPTIONS

SUMMARY OF MAIN POINTS COVERED:

- EXERCISES AND DEFINITIONS RELATED TO EXCEPTIONS

MAIN MESSAGES:

- CHAPTER 11 CONTAINS THE RULES FOR RAISING AND HANDLING EXCEPTIONS

SUBTOPICS:

- IMPORTANT DEFINITIONS
  - EXCEPTION
  - RAISE
  - HANDLING

SPECIAL CONSIDERATIONS:

- CLASS SHOULD BE REMINDED THAT SOME EXCEPTIONS CANNOT BE SUPPRESSED FOR MACHINE DEPENDENT REASONS

EXERCISES:

- ALLOW 25 MINUTES TOTAL
- TWO EXERCISES
  - STUDENTS REQUIRED TO USE LRM CHAPTER 11 TO ANSWER QUESTIONS ABOUT
    - Storage_Error
    - Constraint_Error
    - Program_Error
  - ANSWERS MUST BE SUPPORTED WITH RELEVANT LRM SECTION AND PARAGRAPH NUMBERS

INSTRUCTOR NOTES

- SPECIAL CONSIDERATIONS, LAST BULLET
  - IN SOME SYSTEMS, THE "OBJECT CODE" LIBRARY IS CALLED A PROGRAM LIBRARY, SO THIS MIGHT CONFUSE SOME STUDENTS (INCLUDING THE INSTRUCTORS IN TRAINING).

- EXERCISES
  - EXAMPLES
    - CLASS PRESENTED WITH AN EXAMPLE OF A SUBUNIT OF SUBUNIT WITH THE PARENT UNIT NAME OF THE FORMER NOT GIVEN AS THE REQUIRED FULLY EXPANDED NAME. STUDENTS ASKED TO IDENTIFY THE PROBLEM.
    - CLASS PRESENTED WITH AN EXAMPLE OF A PACKAGE BODY CONTAINING ILLEGAL USE OF SUBPROGRAM FROM PACKAGE INSTANTIATED IN PACKAGE SPECIFICATION. STUDENTS ASKED TO EXPLAIN WHY THIS IS ILLEGAL

SECTION 15 : LRM CHAPTER 10 - PROGRAM STRUCTURE AND COMPILATION ISSUES

SUMMARY OF MAIN POINTS COVERED:

- EXERCISES AND DEFINITIONS RELATED TO PROGRAM STRUCTURE AND COMPILATION

MAIN MESSAGES:

- LRM CHAPTER 10 CONTAINS THE RULES FOR PROGRAM STRUCTURE AND PROGRAM COMPILATION

SUBTOPICS:

- IMPORTANT DEFINITIONS
    - PROGRAM LIBRARY
    - MAIN LIBRARY

SPECIAL CONSIDERATIONS:

- IF SHORT ON TIME, THIS SECTION CAN BE SKIPPED
- ONE OF THE IMPORTANT DEFINITIONS IS A PROGRAM LIBRARY
    - MAKE SURE THE CLASS UNDERSTANDS THAT PROGRAM LIBRARY, AS USED IN Ada, CONTAINS INFORMATION TO SUPPORT SEPARATE COMPILATION
    - THERE IS NO ASSUMPTION THAT THE OBJECT CODE BE PART OF THE LIBRARY
    - SOME IMPLEMENTATIONS MAY COMBINE "OBJECT CODE" LIBRARY WITH THE PROGRAM LIBRARY, BUT THIS IS AN IMPLEMENTATION DESIGN

EXERCISES:

- ALLOW 25 MINUTES TOTAL
- TWO EXERCISES
    - STUDENTS REQUIRED TO USE LRM CHAPTER 10 TO ANSWER QUESTIONS ABOUT
        - PROGRAM STRUCTURE
        - COMPILATION UNITS
    - ANSWERS MUST BE SUPPORTED WITH RELEVANT LRM SECTION AND PARAGRAPH NUMBERS

INSTRUCTOR NOTES

- EXERCISES
  - EXAMPLES
    - STUDENTS PRESENTED WITH AN EXAMPLE OF AN ENUMERATION REPRESENTATION CLAUSE WITH INTEGER CODES NOT IN INCREASING ORDER. STUDENTS ASKED TO EXPLAIN WHY THIS IS ILLEGAL.
    - STUDENTS PRESENTED WITH AN EXAMPLE OF A CODE PROCEDURE CONTAINING LOCAL VARIABLES. STUDENTS ASKED TO EXPLAIN WHY THIS IS ILLEGAL.

SECTION 16 : LRM CHAPTER 13 - REPRESENTATION CLAUSES AND IMPLEMENTATION-DEPENDENT FEATURES

SUMMARY OF MAIN POINTS COVERED:

- EXERCISES DEALING WITH REPRESENTATION CLAUSES AND INTERFACES

MAIN MESSAGES:

- LRM CHAPTER 13 DEALS WITH THE RULES FOR REPRESENTATION CLAUSES AND OTHER IMPLEMENTATION-DEPENDENT FEATURES

EXERCISES:

- ALLOW 25 MINUTES TOTAL
- TWO EXERCISES
    - STUDENTS ARE REQUIRED TO USE LRM CHAPTER 13 TO ANSWER QUESTIONS ABOUT
        - INTERFACING WITH OTHER LANGUAGES
        - REPRESENTATION CLAUSES
        - CODE PROCEDURES
    - ANSWERS MUST BE SUPPORTED WITH RELEVANT LRM SECTION AND PARAGRAPH NUMBERS

INSTRUCTOR NOTES

- THE IMPORTANT DEFINITIONS INCLUDE
  - STATEMENT - AN ACTION TO BE PERFORMED
  - EXECUTION - THE PROCESS BY WHICH A STATEMENT ACHIEVES ITS ACTION

- EXERCISES
  - EXAMPLES
    - STUDENTS PRESENTED WITH AN EXAMPLE OF A CASE STATEMENT WITH A CHOICE THAT IS NOT A SIMPLE EXPRESSION. STUDENTS ASKED TO EXPLAIN WHY THIS IS ILLEGAL
    - STUDENTS PRESENTED WITH AN EXAMPLE OF A GOTO STATEMENT ATTEMPTING TO BRANCH OUTSIDE OF THE BLOCK CONTAINING IT. STUDENTS ASKED TO EXPLAIN WHY THIS IS ILLEGAL

SECTION 17 : LRM CHAPTER 5 - STATEMENTS

SUMMARY OF MAIN POINTS COVERED:

- EXERCISES AND DEFINITIONS RELATED TO STATEMENTS

MAIN MESSAGES:

- LRM CHAPTER 5 CONTAINS THE RULES FOR STATEMENTS

SUBTOPICS:

- IMPORTANT DEFINITIONS
  - STATEMENT
  - EXECUTION

SPECIAL CONSIDERATIONS:

- THIS CHAPTER IS FAIRLY STRAIGHTFORWARD AND MAY BE SKIPPED IF TIME IS SHORT

EXERCISES:

- ALLOW 25 MINUTES TOTAL
- TWO EXERCISES
  - STUDENTS ARE REQUIRED TO USE LRM CHAPTERS 3, 4 AND 5 TO ANSWER QUESTIONS ABOUT
    - CASE STATEMENTS
    - ASSIGNMENTS STATEMENTS
    - LOOP STATEMENTS
    - GOTO STATEMENTS
  - ANSWERS MUST BE SUPPORTED WITH RELEVANT LRM SECTION AND PARAGRAPH NUMBERS

INSTRUCTOR NOTES

- THE IMPORTANT DEFINITIONS INCLUDE EXTERNAL FILE AND FILE

- EXERCISES

    - FIRST EXAMPLE: THE EXAMPLE CONTAINS THE STATEMENTS

```
if not End_Of_File (F) then
   Get (X);
end if;
```

BECAUSE THE TERMINATORS "EXIST", THE FIRST IS NOT EMPTY. HOWEVER, THE GET SKIPS PAST THE LINE AND PAGE TERMINATORS, SO NO DATA TO READ.

SECTION 18 : LRM CHAPTER 14 - INPUT/OUTPUT

SUMMARY OF MAIN POINTS COVERED:

- EXERCISES AND DEFINITIONS RELATED TO INPUT/OUTPUT

MAIN MESSAGES:

- LRM CHAPTER 14 CONTAINS THE RULES FOR I/O IN Ada

SUBTOPICS:

- IMPORTANT DEFINITIONS
  - EXTERNAL FILE
  - FILE
  - OPEN/CLOSED

EXERCISES:

- ALLOW 55 MINUTES TOTAL
- TWO EXERCISES
  - STUDENTS ARE REQUIRED TO USE LRM CHAPTER 14 TO ANSWER QUESTIONS ABOUT
    - LINE, PAGE AND FILE TERMINATORS
    - INSTANTIATIONS OF Sequential_IO
    - Reset
    - OPENING AND CLOSING FILES
  - EXAMPLES
    - STUDENTS PRESENTED WITH AN EXAMPLE OF FILE CONTAINING ONLY TERMINATORS, THAT RESULTS IN End_Error BEING RAISED ON THE FIRST GET. STUDENTS ASKED TO EXPLAIN WHY
    - STUDENTS ASKED TO EXPLAIN IF IT IS LEGAL TO INSTANTIATE Sequential_IO WITH String AS GENERIC ACTUAL PARAMETER
  - ANSWERS MUST BE SUPPORTED WITH RELEVANT LRM SECTION AND PARAGRAPH NUMBERS

Material: Instructor's Course S500 Advanced Language Modules

We would appreciate your comments on this material and would like you to complete this brief questionaire. The completed questionaire should be forwarded to the address on the back of this page. Thank you in advance for your time and effort.

1. Your name, company or affiliation, address and phone number.

2. Was the material accurate and technically correct?

   Yes □ No □

   Comments:

3. Were there any typographical errors?

   Yes □ No □

   If yes, on what pages?

4. Was the material organized and presented appropriately for your applications?

   Yes □ No □

   Comments:

5. General Comments:

place
stamp
here

COMMANDER
US ARMY MATERIEL COMMAND
ATTN: AMCDE-SB (OGLESBY)
5001 EISENHOWER AVENUE
ALEXANDRIA, VIRGINIA 22233

END
DTIC
FILMED
4-86